
PROJECT BIRD
(Título provisional)
DOCUMENTO DE DISEÑO
Versión 1.0 - 25 de Febrero de 2021
Sunlight
STUDIO

# ÍNDICE

# INTRODUCCIÓN

Este documento forma parte del material empleado para desarrollar *"Project Bird"*, un videojuego musical multiplataforma de *Sunlight Studio*.

En él veremos una serie de descripciones detalladas sobre los diferentes apartados con los que deberá contar el título para ser considerado un producto completo y competente, facilitando así su desarrollo.

*"Project Bird"* es un videojuego que se encuentra en plena etapa de conceptualización y desarrollo, por lo que tanto este documento como todos los recursos mostrados en él están sujetos a cambios.











Todo el material mostrado (a excepción de indicar lo contrario) es propiedad intelectual de los miembros de *Sunlight Studio* listados a continuación:

- Proyecto diseñado, dirigido y desarrollado por **Juan Manuel Ruiz Culiáñez**.















- Arte 2D de escenarios y personajes diseñado por **Boris García Pérez**.









- Música para niveles, menús e interfaces compuesta por **José Sala Lozano**.





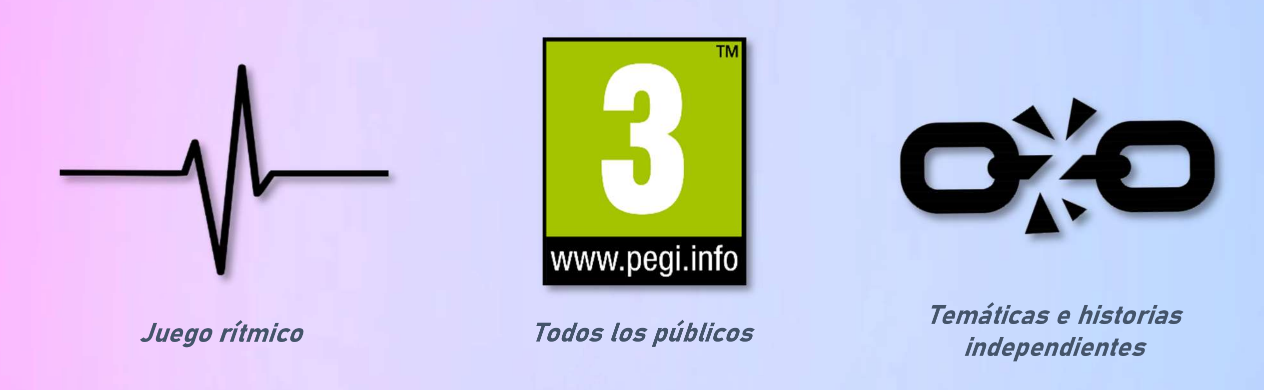



# DESCRIPCIÓN DEL VIDEOJUEGO

## ▪ ¿QUÉ ES PROJECT BIRD?

*Project Bird* es un videojuego rítmico para todos los públicos en el que tendremos que completar una serie de minijuegos con diversos personajes, melodías, temáticas e historias.

Cada minijuego será completamente independiente del resto, de forma que podamos paralelizar la estética, temática y musicalidad de cada uno.




*Juego rítmico* | *Todos los públicos* | *Temáticas e historias independientes*

Actualmente el proyecto se encuentra en una etapa muy temprana de desarrollo, por lo que hemos decidido centrarnos en la creación de un único minijuego como *MVP* (Producto Mínimo Viable).

Para este primer minijuego, nuestro protagonista será un pájaro que va picando al ritmo de la música. De esta manera, decidimos titular provisionalmente al videojuego como "*Project Bird*".



## ▪ PUNTOS CLAVE Y CARACTERÍSTICAS

Nuestro objetivo con *Project Bird* es crear un videojuego con acabado profesional pese a todas las limitaciones impuestas por falta de personal o problemas relacionados con la actual crisis sanitaria por Covid-19.

De esta manera y teniendo en cuenta todos los puntos fuertes de cada uno de los integrantes del equipo, hemos decidido llevar a cabo el desarrollo de un proyecto que cumpla las siguientes características:

### 1. Estética *"cartoon"* cuidada y amigable.



Con el objetivo de hacer el juego más accesible para todos los públicos y que resulte visualmente agradable, hemos elegido una estética *"cartoon"* 2D con la que mostraremos escenarios y protagonistas desenfadados.

### 2. Mini historias independientes (formato).



Para hacer más sencillo el desarrollo del título, cada minijuego será completamente independiente de otro en cuanto a escenarios, música y/o personajes.

Podrán reciclarse ciertos assets, pero no será lo habitual.

### 3. Melodías originales y pegadizas.



Contamos con un compositor musical dispuesto a crear desde cero los todos los sonidos y temas que necesite el videojuego.

## 4. Acción con un único botón.



Toda la acción se controlará mediante un único botón.
Esto facilita el aprendizaje y uso, además de permitir portar el título a posibles nuevas plataformas.

## 5. Minijuegos cortos y rejugables.



Cada minijuego tendrá una duración aproximada de 1,5 ~ 2 minutos y marcadores de puntuación para incentivar la rejugabilidad.

# ▪ REFERENTES

Los referentes en materia de videojuegos, estética y música que vamos a tener en consideración van a ser los siguientes:

## ▪ REFERENTES JUGABLES




*Rhythm Paradise (GBA, NDS, WII, 3DS)*




*Captura (Gameplay Video)*

## ¿Qué destacamos de "*Rhythm Paradise*"?

1. *Feedback* visual y sonoro.
2. Formato de sus minijuegos.




*Rhythm Doctor (PC, Switch, Android, IOS)*



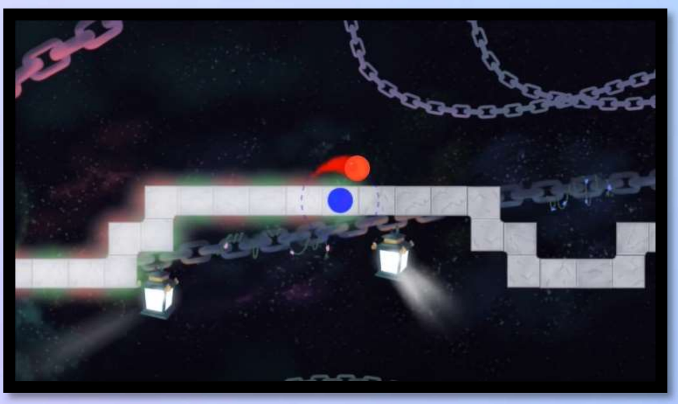

*Captura (Gameplay Video)*

## ¿Qué destacamos de "*Rhythm Doctor*"?

1. Interfaz de usuario (*UI / UX*).
2. Tutorial y configuración inicial.




*A Dance of Fire and Ice (PC, Android, IOS)*



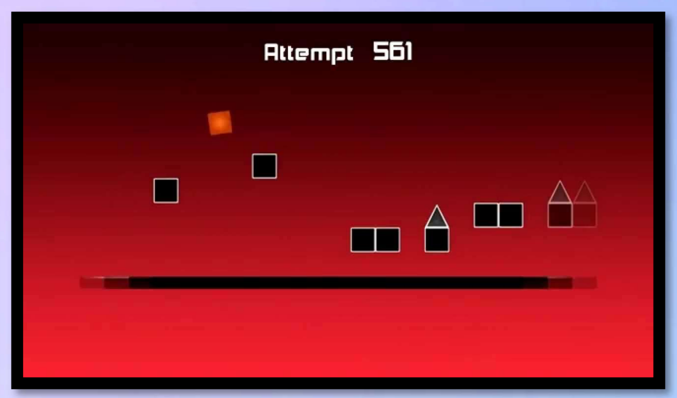
*Captura (Gameplay Video)*

## ¿Qué destacamos de "*A Dance of Fire and Ice*"?

1. Control mediante un único botón.
2. Estructuras musicales (*BPM's* y progresiones).




*The Impossible Game (PC, Android, IOS, PS3, X360)*




*Captura (Gameplay Video)*

## ¿Qué destacamos de "*The Impossible Game*"?

1. Implementación de diferentes mecánicas pese a usar un único botón.
2. Modelo de negocio.




*Super Hexagon (PC, Android, IOS, PS3, X360)*

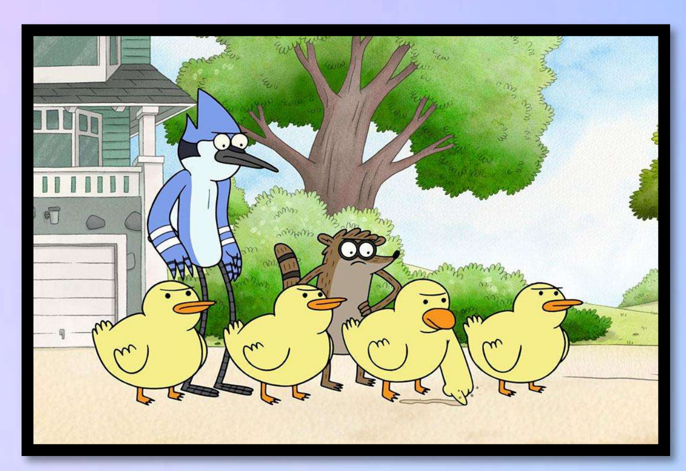



*Captura (Gameplay Video)*

## ¿Qué destacamos de "*Super Hexagon*"?

1. Integración entre menú y niveles.
2. Música pegadiza y animada.

## ▪ REFERENTES ESTÉTICOS

*Regular Show (Trailer)*




*Bojack Horseman (Trailer)*





*Over the Garden Wall (Trailer)*






*Final Space (Trailer)*

## ▪ REFERENTES MUSICALES




*C2C*




*C2C – Down the Road (Music Video)*




*C2C – Delta (Music Video)*




*Caravan Palace (CP)*



*CP – Lone Digger (Music Video)*



*CP – Rock It for Me (Music Video)*

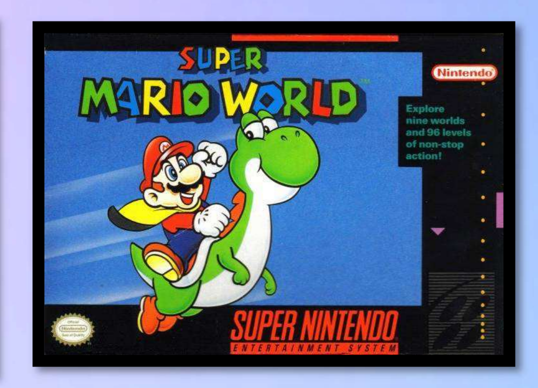




*Koji Kondo (KK)*




*KK – Super Mario Land Main Theme ([Music Video](#))*

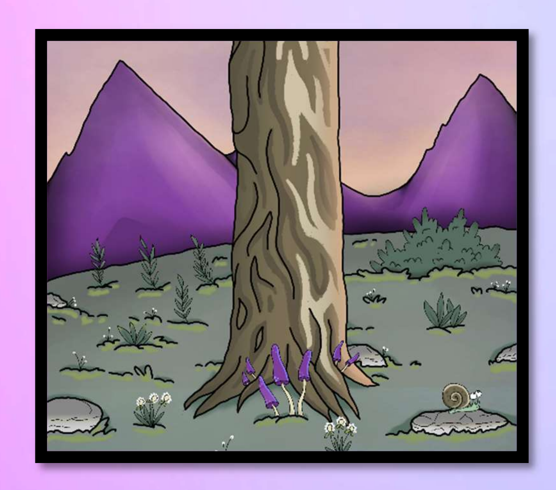



*KK – Super Mario World Main Theme ([Music Video](#))*

## ▪ TONO Y ESTILO ADOPTADO

El videojuego será 2D con estética *cartoon*, teniendo muy en cuenta los referentes citados anteriormente para la elaboración de mecánicas, personajes, escenarios, música, etc.

Cada minijuego será independiente del resto, de forma que en muchas ocasiones resultará necesario añadir una cinemática introductoria y final para poder situar al jugador.

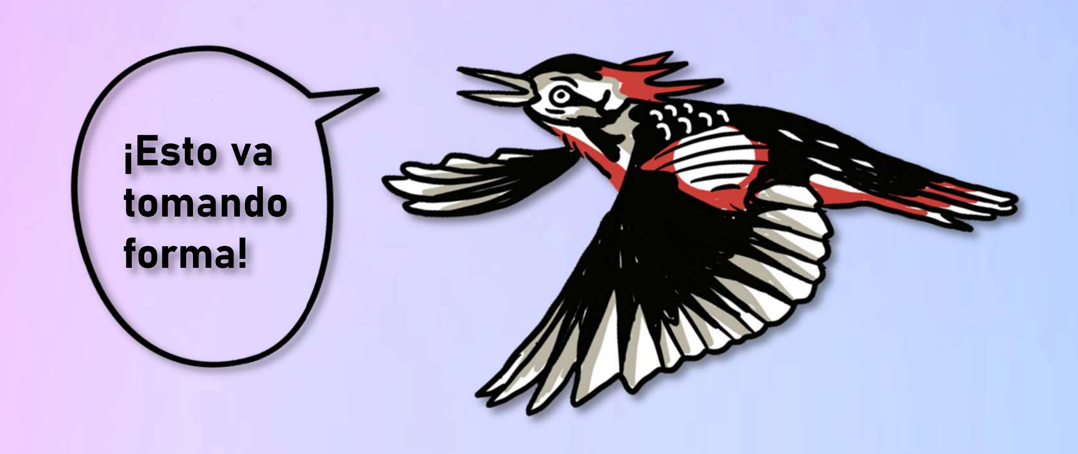




*Diseño y personajes del minijuego en desarrollo*

El tono general del mismo será desenfadado, con la presencia de toques humorísticos y una música que refuerce el carácter satírico general.

Las historias contarán situaciones caricaturizadas con el fin de justificar la acción del minijuego en cuestión.




# HISTORIA(S)

## ▪ ESTRUCTURA GENERAL DE NIVELES

La estructura general que seguirán los minijuegos será la siguiente:

### 1. Cinemática introductoria

Al comenzar, todo minijuego tendrá que poner en situación al jugador mediante una cinemática introductoria.

Su duración será breve (10 ~ 15 segundos), y servirá también para comenzar a reproducir la música.

Puede omitirse manteniendo pulsado el botón de acción.

### 2. *Gameplay*

Tras finalizar la cinemática, aparecerá en primer plano la escena del juego, donde se mostrará la interfaz *in-game* y el jugador podrá controlar al personaje.

Su duración ocupará la mayor parte del tiempo (1,2 ~ 1,5 minutos).

Puede ser reintentado desde el menú de pausa sin tener que volver a ver la cinemática inicial.

### 3. (Opcional) Interludios / Transiciones

Algunos de los minijuegos tendrán que interrumpir la acción momentáneamente para permitir ciertos cambios musicales y de escenario. Para ello, se emplearán animaciones auxiliares.

Su duración será extremadamente breve (2 ~ 3 segundos).

### 4. Cinemática final

Tras finalizar la acción, aparecerá otra animación para cerrar la historia.

Su duración será breve (10 ~ 15 segundos), y servirá también para acompañar al cierre de la música.

Puede omitirse manteniendo pulsado el botón de acción.

## ▪ PRESENTACIÓN DE LAS HISTORIAS

Para comprobar que todos los minijuegos siguen la estructura de niveles descrita anteriormente, vamos a diseccionarlos uno a uno:

## ▪ Minijuego 1: Pájaro Hambriento

### 1. Cinemática introductoria



Un ave vuela raso por una zona montañosa buscando algo de alimento.

Al ver un gusano salir de la corteza de un árbol, cambia el rumbo de su vuelo realizando una acrobacia y se posa sobre el mismo.

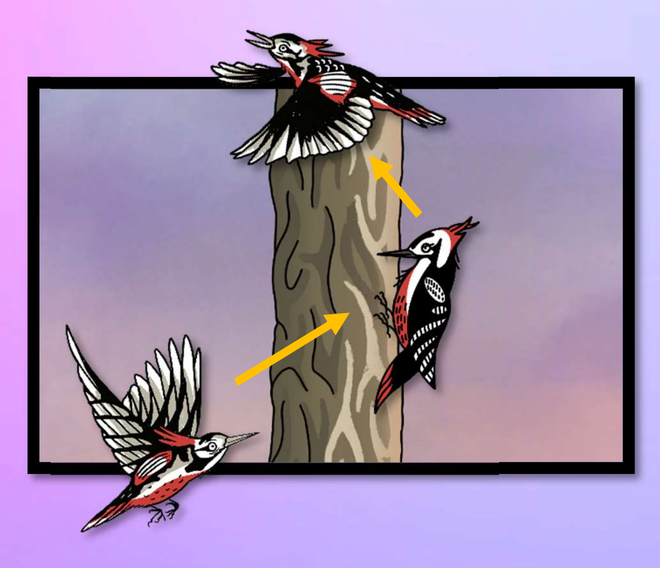
### 2. Gameplay [Pt. 1]



El pájaro tratará de picar el árbol para obtener el gusano. Este picoteo será manejado por el jugador.

Deberemos pulsar el botón de acción en el momento exacto para que pique correctamente.

### 3. Transición [1]



Tras haber comido el primer gusano, el pájaro cambiará de posición para encontrar más alimento.

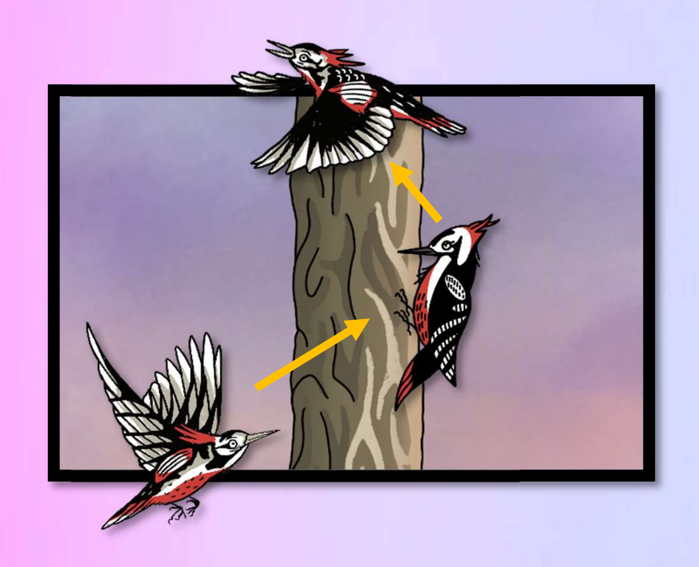
## 4. Gameplay [Pt. 2]



El pájaro tendrá que volver a picar para conseguir otro gusano. Este picoteo será manejado por el jugador.

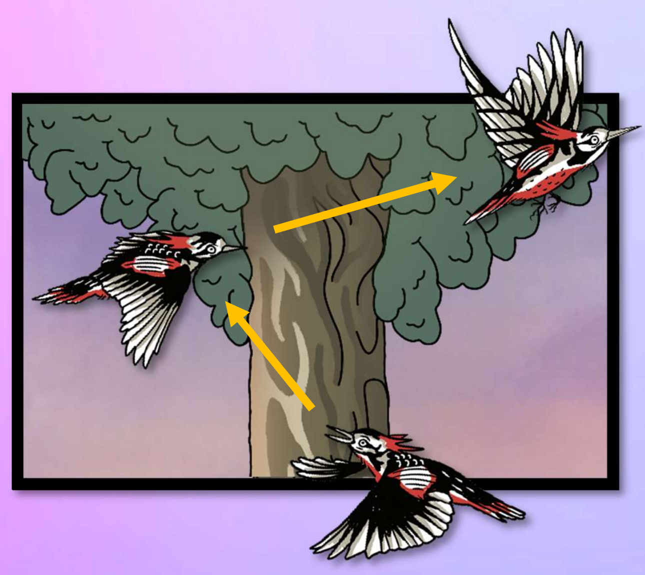
Deberemos pulsar el botón de acción en el momento exacto para que pique correctamente.

## 5. Transición [2]



Tras haber comido el segundo gusano, el pájaro cambiará su posición de nuevo para introducir la cinemática final.

## 6. Cinemática final



Tras la última transición, el pájaro habrá llegado a la copa del árbol y alzará el vuelo.

Termina el minijuego.

# JUGABILIDAD Y CÁMARA

## ▪ OBJETIVOS GENERALES

Los objetivos generales del juego listados por orden de prioridad son los siguientes:

1. Completar todos los niveles.
2. Obtener la puntuación más alta en cada nivel (*Overall score*).
3. Conseguir la calificación más alta en cada nivel. *(F, D, C, B, A, S).*
4. Mantener la racha de buenas calificaciones más alta posible (a mayor racha, mayor multiplicador de puntos).
5. Competir por obtener la puntuación global más alta.

## ▪ MECÁNICAS DE JUEGO

Todos los minijuegos consistirán en seguir el ritmo de la canción correspondiente pulsando un único botón.

Tanto la acción del jugador como los movimientos del personaje irán en todo momento condicionados por el ritmo de la música, por lo que se deberá seguir el ritmo de la forma más precisa posible.

La música deberá aumentar de ritmo progresivamente, de forma que también suba la dificultad del juego y esté correctamente balanceado.

**BPM = DIFICULTAD**

*(Beats Por Minuto)*

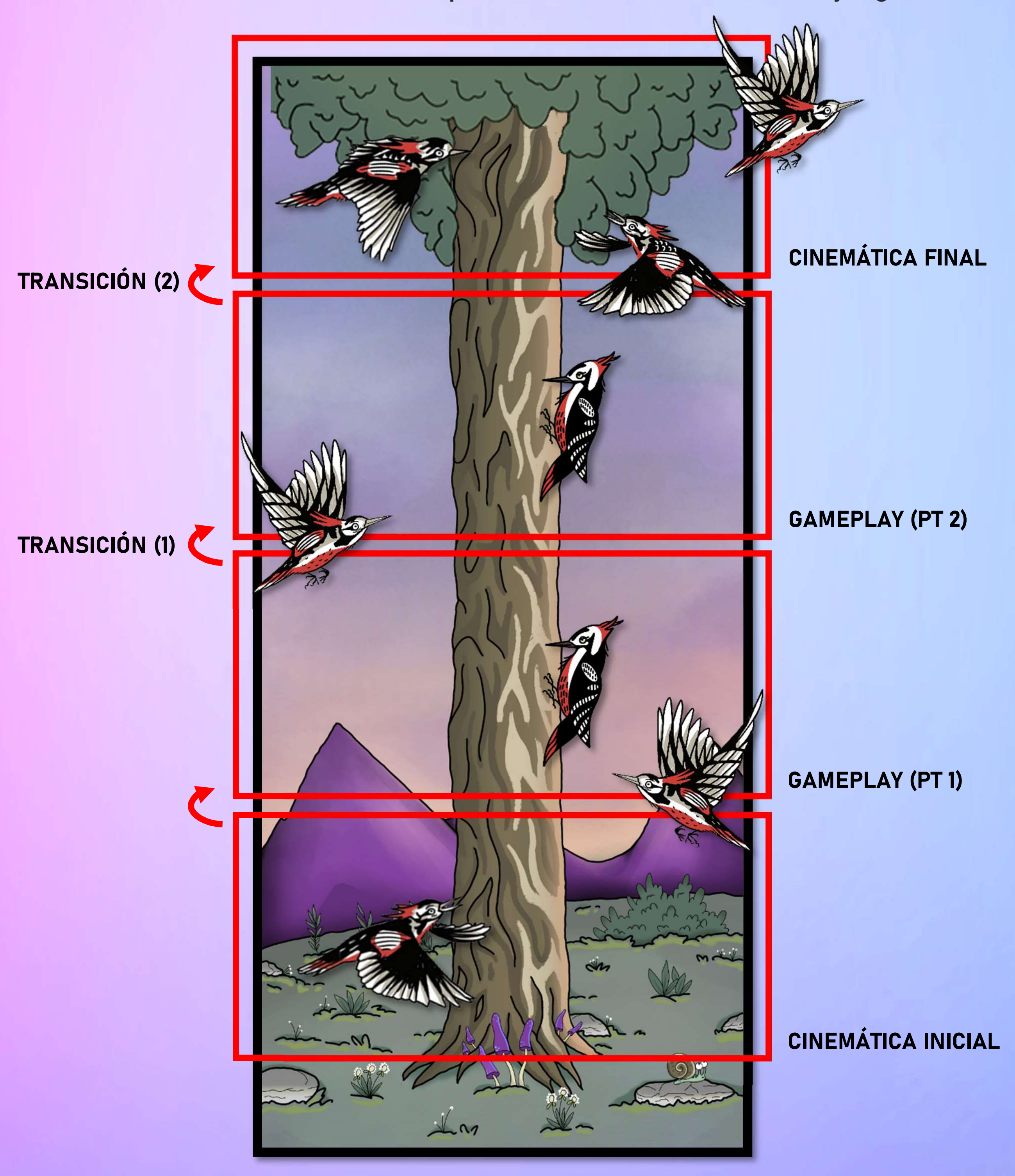
Tras terminar el nivel, se nos proporcionará una calificación que nos permitirá o no jugar al siguiente, además de una puntuación.

## ▪ FUNCIONAMIENTO DE LA CÁMARA

La cámara será estática, por lo que no podrá ser controlada por el jugador y se dedicará únicamente a ofrecer un primer plano del personaje.

Para compensar la monotonía de este tipo de cámara, aparecerán estímulos constantes en la escena mediante la implementación de un efecto "*parallax*" por capas.

Veamos el funcionamiento completo de la cámara en un minijuego:



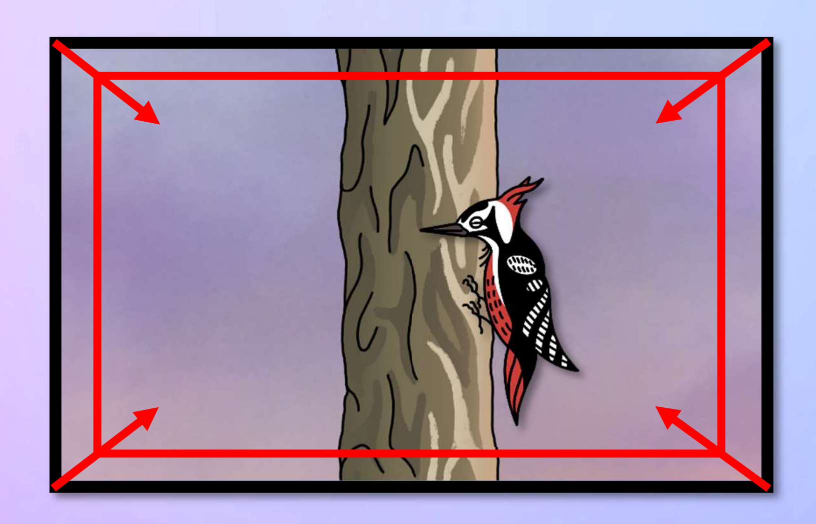
## ▪ EFECTOS DE CÁMARA

### ▪ *ZOOM OUT*



La cámara hará automáticamente un efecto "*Zoom Out*" del 1~2% en función de los BPM (*Beats* Por Minuto) de la canción.

### ▪ *ZOOM IN*



La cámara hará automáticamente un efecto "*Zoom In*" del ~2% cada vez que el jugador pulse el botón de acción.

# ▪ DIAGRAMA DE FLUJO

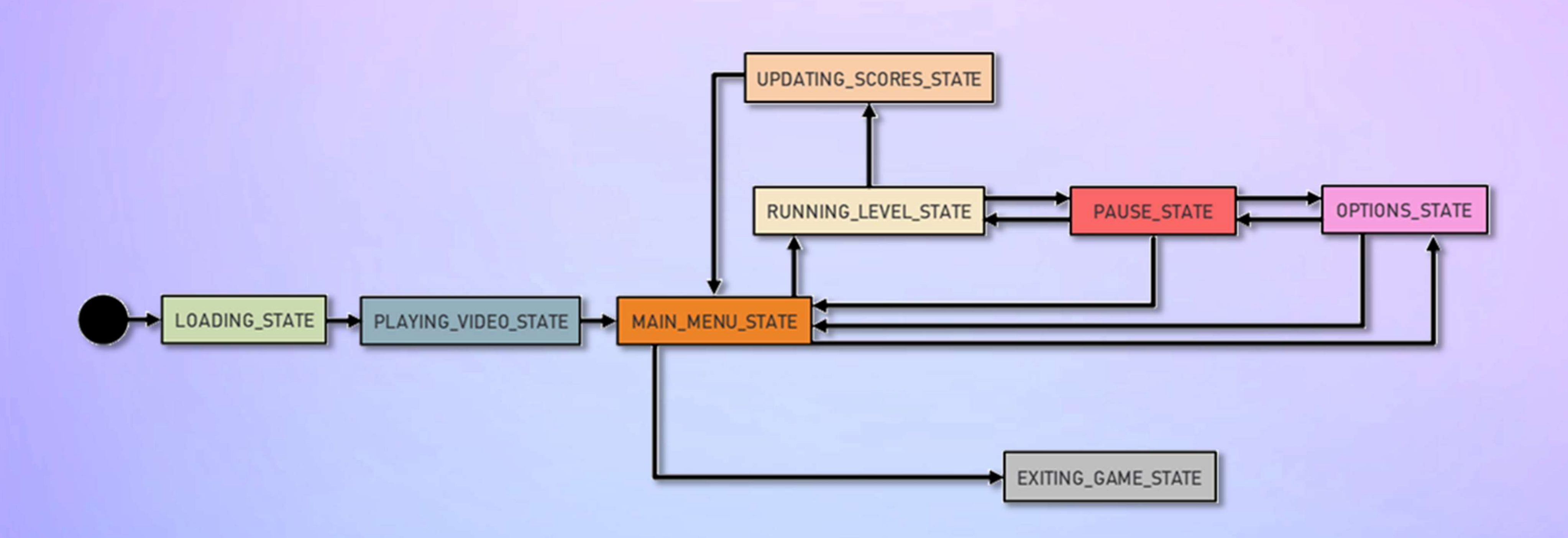
## ▪ ESTADOS DE JUEGO

Los estados con los que contará el juego son los siguientes:

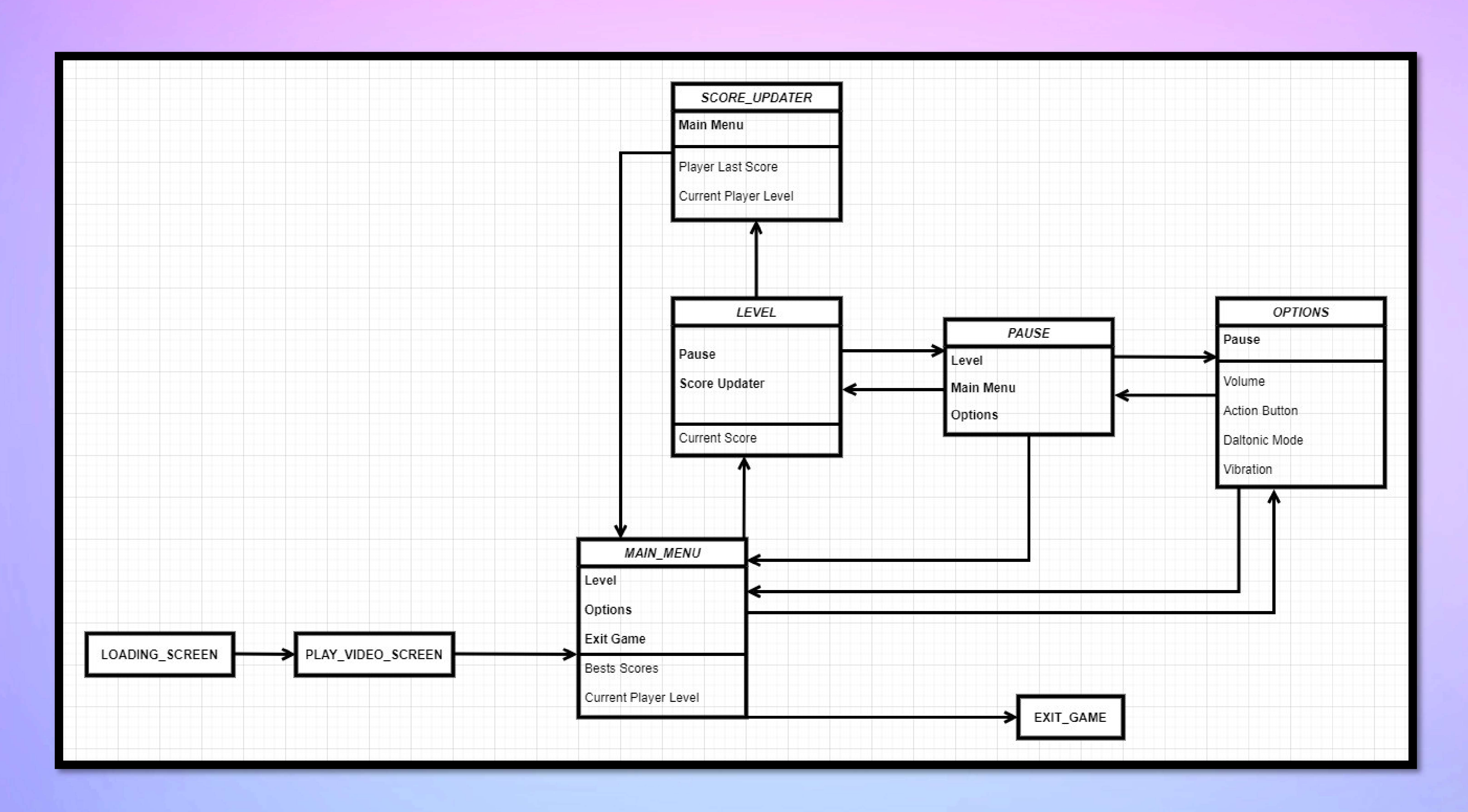
1. PLAYING_VIDEO_STATE : Reproducción de video.
2. LOADING_STATE : Carga de datos.
3. MAIN_MENU_STATE : Estancia en menú principal.
4. RUNNING_LEVEL_STATE : Ejecución de un nivel.
5. PAUSE_STATE : Estancia en menú de pausa.
6. OPTIONS_STATE : Estancia en menú de opciones.
7. UPDATING_SCORES_STATE : Actualización de las puntuaciones.
8. EXITING_GAME_STATE : Saliendo del videojuego.

Una vez definidos todos los estados de juego posibles, procedemos a crear los siguientes diagramas:

- **DIAGRAMA DE ESTADOS DE JUEGO**



## ▪ DIAGRAMA DE ESTADOS DE JUEGO AVANZADO



# ▪ SISTEMA DE PUNTUACIONES

## ▪ TIPOS DE PUNTUACIONES

Estos son los diferentes tipos de puntuaciones con los que contará el juego:

1. **Puntuación individual de nivel (Score).**

La puntuación que consigamos en un nivel dependerá de la precisión con la que el jugador siga el ritmo de la música.

Esta es la tabla de puntuaciones según la precisión:

| **Tabla de puntuaciones** | |
|---|---|
| *Timing* (Precisión) | Puntuación |
| Perfecto (±0 cs) | 50 puntos |
| Bien (±10 cs) | 30 puntos |
| Regular (±15 cs) | 10 puntos |

**NOTA:** cs = Centésimas de segundo.

Para obtener la puntuación final (PF), deberemos de sumar todos los puntos (PE) y multiplicarlos por el multiplicador de puntos que tenga el jugador:

**Puntuación Final (PF) = Puntuación Estándar (PE) * Multiplicador**

## 2. Calificación individual de nivel (*F, D, C, B, A, S*).

En función de la puntuación obtenida en un nivel (puntuación estándar (PE), no final (PF)), se otorgará una calificación para orientar al jugador.

Debido a que cada minijuego variará en aspectos de duración y dificultad, no podemos definir una tabla de calificaciones general.

Veamos como ejemplo la tabla de calificaciones para el minijuego "Pájaro Hambriento":

| **Puntuaciones minijuego "Pájaro Hambriento"** | | | |
|---|---|---|---|
| Duración (s) | *Beats* a seguir por el jugador | Puntuación estándar (PE) MÁXIMA | Puntuación estándar (PE) MEDIA |
| ~ 90 s | 64 Beats | 3.200 puntos | ~1.540 puntos |

**NOTA:**

- **Puntuación estándar MÁXIMA =**
  - **Perfecto:** 64 (3.200 puntos)

- **Puntuación estándar MEDIA =**
  - **Perfecto:** 14 (600 puntos)
  - **Bien:** 26 (780 puntos)
  - **Regular:** 16 (160 puntos)
  - **Fallos:** 8 (0 puntos)

Una vez hayamos obtenido la puntuación estándar, podemos calcular la calificación correspondiente:

| Tabla calificaciones minijuego “Pájaro Hambriento” | |
|---|---|
| Calificación | Puntos requeridos |
| F (Insuficiente) | 0 puntos |
| D (Insuficiente) | +220 puntos |
| C (Suficiente) | +480 puntos |
| B (Bien) | +1050 puntos |
| A (Notable) | +2100 puntos |
| S (Sobresaliente) | +2650 puntos |
| *Perfect* (Perfecto) | 3.200 puntos |

### 3. Multiplicador de puntuaciones.

La calificación que obtengamos en cada nivel nos puede permitir aumentar el multiplicador empleado para calcular la puntuación final.

El multiplicador tendrá por defecto un valor de **x1.0** y aumentará conforme el jugador mantenga buenas calificaciones o las aumente.

La tabla del multiplicador es la siguiente:

| Multiplicador de puntuaciones | |
|---|---|
| Última calificación | Nuevo multiplicador |
| F (Insuficiente) | x 0.0 |
| D (Insuficiente) | x 0.0 |
| C (Suficiente) | x 1.0 |
| B (Bien) | x 1.3 |
| A (Notable) | x 1.6 |
| S (Sobresaliente) | x 2 |
| *Perfect* (Perfecto) | x 2.5 |
| Misma Calificación | Mismo Multiplicador |

El multiplicador se reiniciará cuando la calificación obtenida sea más baja que la anterior.

### 4. Puntuación general del perfil (*Overall score*).

El jugador también contará con una puntuación personal que irá aumentando conforme más minijuegos complete.

Servirá de utilidad para que el jugador sea consciente de su nivel de experiencia.

Cada vez que finalice un nivel, se dividirán los puntos obtenidos entre 100 y se sumarán a la puntuación actual del jugador.

**Puntuación Jugador (PJ) = PJ + Puntuación Final (PF)/100**

## ▪ CONTROLES

### ▪ CONTROL POR MANDO







**NOTA:** Mismos controles para mandos de otras plataformas.

Teniendo en cuenta que todos los periféricos tienen "*Input Lag*" (latencia), es necesario aplicar una corrección de latencia al mando que vayamos a usar.

Basándonos en un estudio realizado por *Rocket Science*, la latencia de los mandos más usados es el siguiente:

| ***Input Lag*** **(latencia) en mandos** | |
|---|---|
| Mando | Latencia (ms) |
| DualShock 4 | 13.3 ms |
| Steam Controller | 16.5 ms |
| Xbox One Controller | 17.4 ms |
| Xbox 360 Controller | 18.8 ms |
| DualShock 3 | 24.1 ms |

**NOTA:** cs = Centésimas de segundo. ms = Milésimas de segundo.

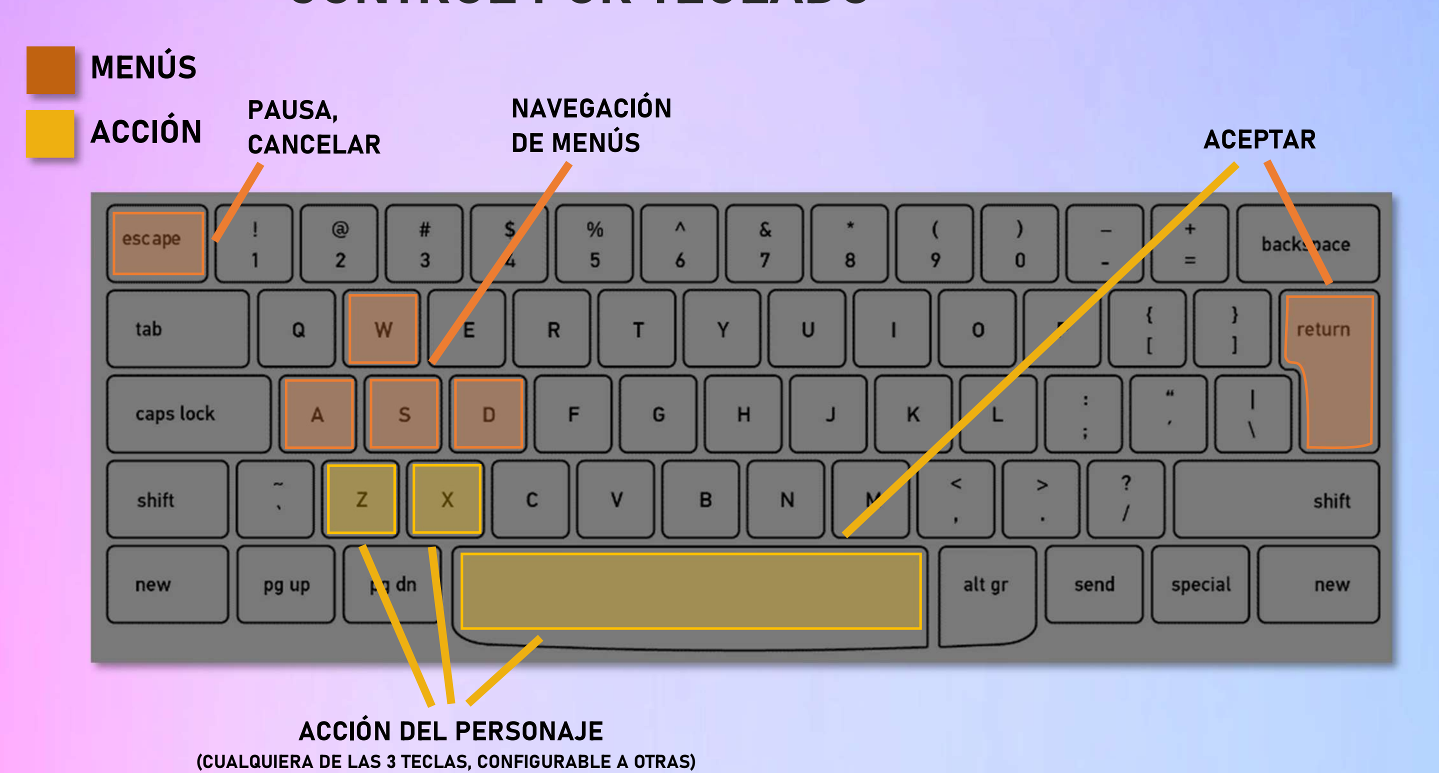
Para calcular la correspondiente corrección, es necesario emplear la siguiente ecuación:

**CORRECCIÓN INPUT =** ***Timing* + *Input Lag***

Aplicamos esta corrección en el DualShock 4 y Xbox One Controller:

| ***Corrección DualShock 4 y Xbox One Controller*** | | |
|---|---|---|
| Mando | *Timing* | Corrección (cs) |
| DualShock 4 | Perfecto (±0 cs) | 0 cs + 13.3 ms = **1.33 cs** |
| - | Bien (±10 cs) | 10 cs + 13.3 ms = **11.33 cs** |
| - | Regular (±15 cs) | 15 cs + 13.3 ms = **16.33 cs** |
| Xbox One Controller | Perfecto (±0 cs) | 0 cs + 17.4 ms = **1.74 cs** |
| - | Bien (±10 cs) | 10 cs + 17.4 ms = **11.74 cs** |
| - | Regular (±15 cs) | 15 cs + 17.4 ms = **16.74 cs** |

## ▪ CONTROL POR TECLADO




Los teclados (estándar y *gaming*) también sufren de input lag. Según [duanluu.com](duanluu.com), las latencias medias en un teclado son las siguientes:

| *Input Lag* (latencia) en teclados | |
|---|---|
| Teclado | Latencia (ms) |
| Teclado estándar USB (media) | 55 ms |
| Teclado *gaming* USB (media) | 25 ms |

**NOTA:** cs = Centésimas de segundo. ms = Milésimas de segundo.

Para calcular la correspondiente corrección, es necesario emplear la siguiente ecuación:

**CORRECCIÓN INPUT = *Timing* + *Input Lag***

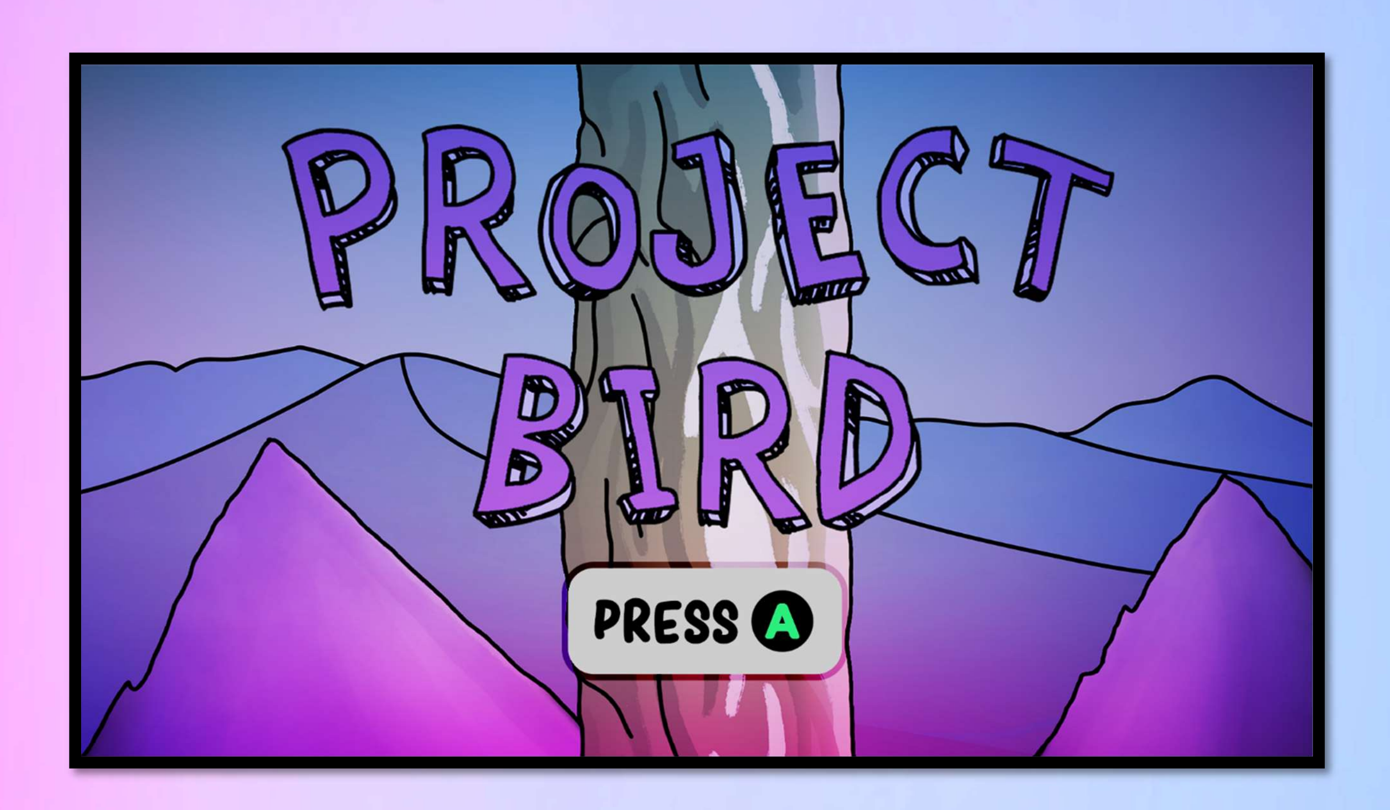
Aplicamos esta corrección en ambos tipos de teclado:

| ***Corrección teclado estándar y teclado gaming*** | | |
|---|---|---|
| Teclado | *Timing* | Corrección (cs) |
| Estándar USB (media) | Perfecto (±0 cs) | 0 cs + 55 ms = **5.50 cs** |
| - | Bien (±10 cs) | 10 cs + 55 ms = **15.50 cs** |
| - | Regular (±15 cs) | 15 cs + 55 ms = **20.50 cs** |
| *Gaming* USB (media) | Perfecto (±0 cs) | 0 cs + 25 ms = **2.50 cs** |
| - | Bien (±10 cs) | 10 cs + 25 ms = **12.50 cs** |
| - | Regular (±15 cs) | 15 cs + 25 ms = **17.50 cs** |



# INTERFAZ

## ▪ PANTALLA DE INICIO

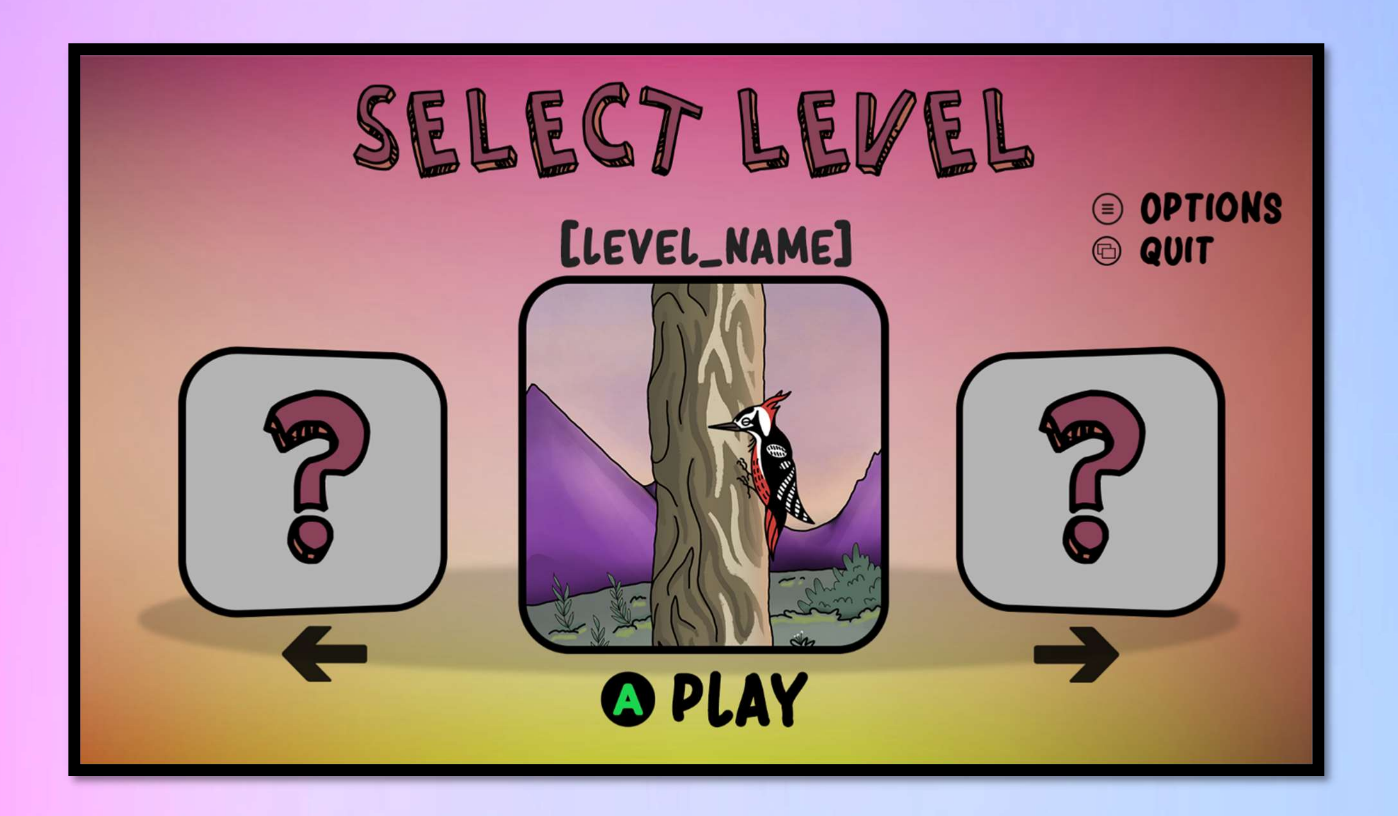



## ▪ ELEMENTOS DE LA PANTALLA DE INICIO

1. **Logo del videojuego.**
   Logo provisional del videojuego.

2. **Llamada al jugador para que presione un botón.**
   Tras pulsar el botón, redirigirá al menú principal.

## ▪ MENÚ PRINCIPAL




## ▪ ELEMENTOS DEL MENÚ PRINCIPAL

1. **Título del menú.**
   "*Select Level*".

2. **Iconos de navegación de menú.**
   Flechas, botones de "*options*", "*quit*" y "*play*".

3. **Información del nivel seleccionado.**
   Icono resaltado. Nombre del minijuego.

4. **Lista de niveles disponibles.**
   Iconos visibles al fondo, detrás de la selección actual.

# ▪ INTERFAZ *IN-GAME*




## ▪ ELEMENTOS DE LA INTERFAZ IN-GAME

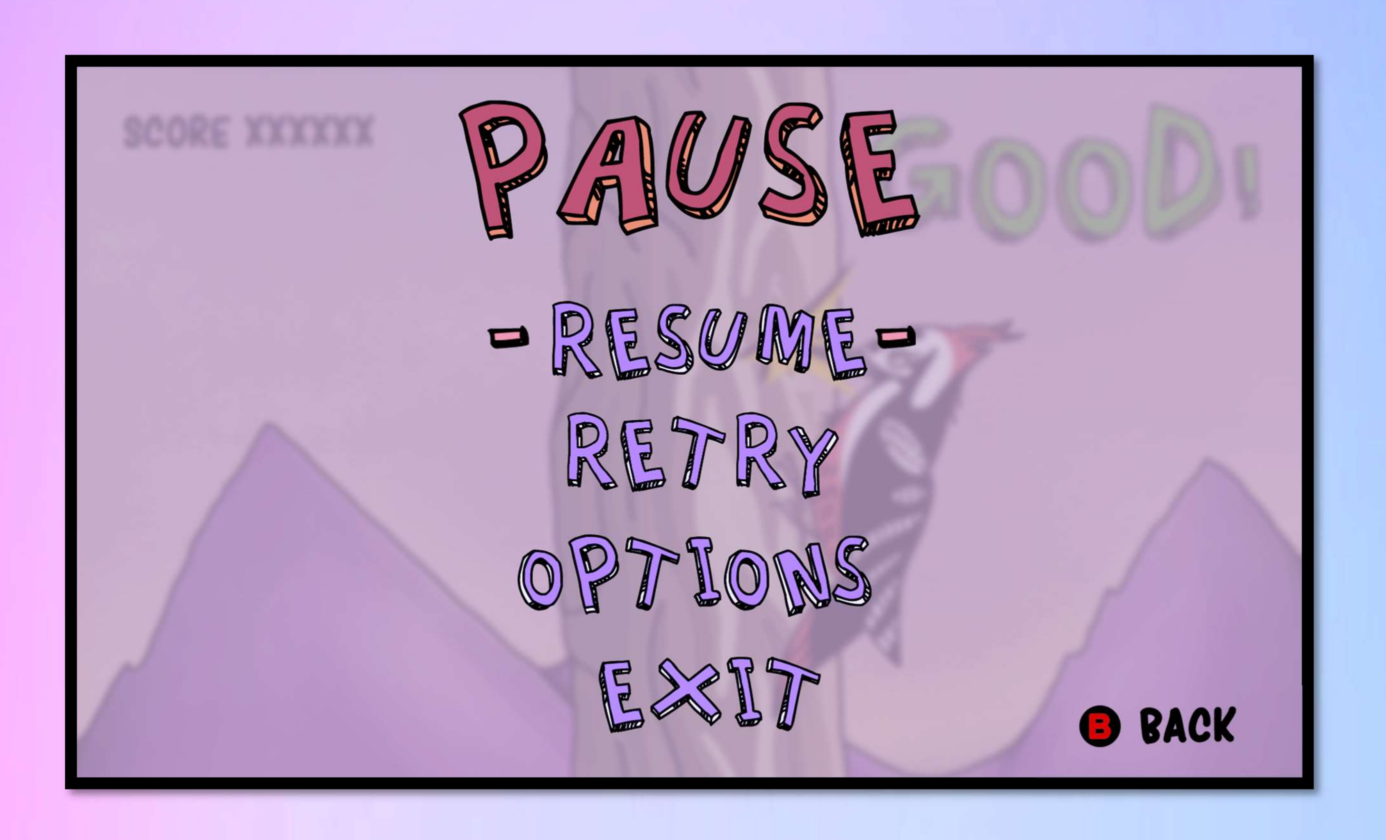
1. **Escenario y personaje.**
   Lugar donde sucede la acción del juego.
   Personaje controlable por el jugador.

2. **Efecto de acción.**
   Efecto en pantalla que cuando se pulse el botón de acción.
   En esta imagen es una chispa de color amarillo.

3. **Feedback de última pulsación.**
   Nos indica si hemos acertado nuestra última acción.
   "*Perfect*", "*Good*", "*OK*" y "*Bad*".

4. **Contador de puntación.**
   Puntuación obtenida durante la partida.

## ▪ MENÚ DE PAUSA




### ▪ ELEMENTOS DEL MENÚ DE PAUSA

1. **Título del menú.**
   "*Pause*".

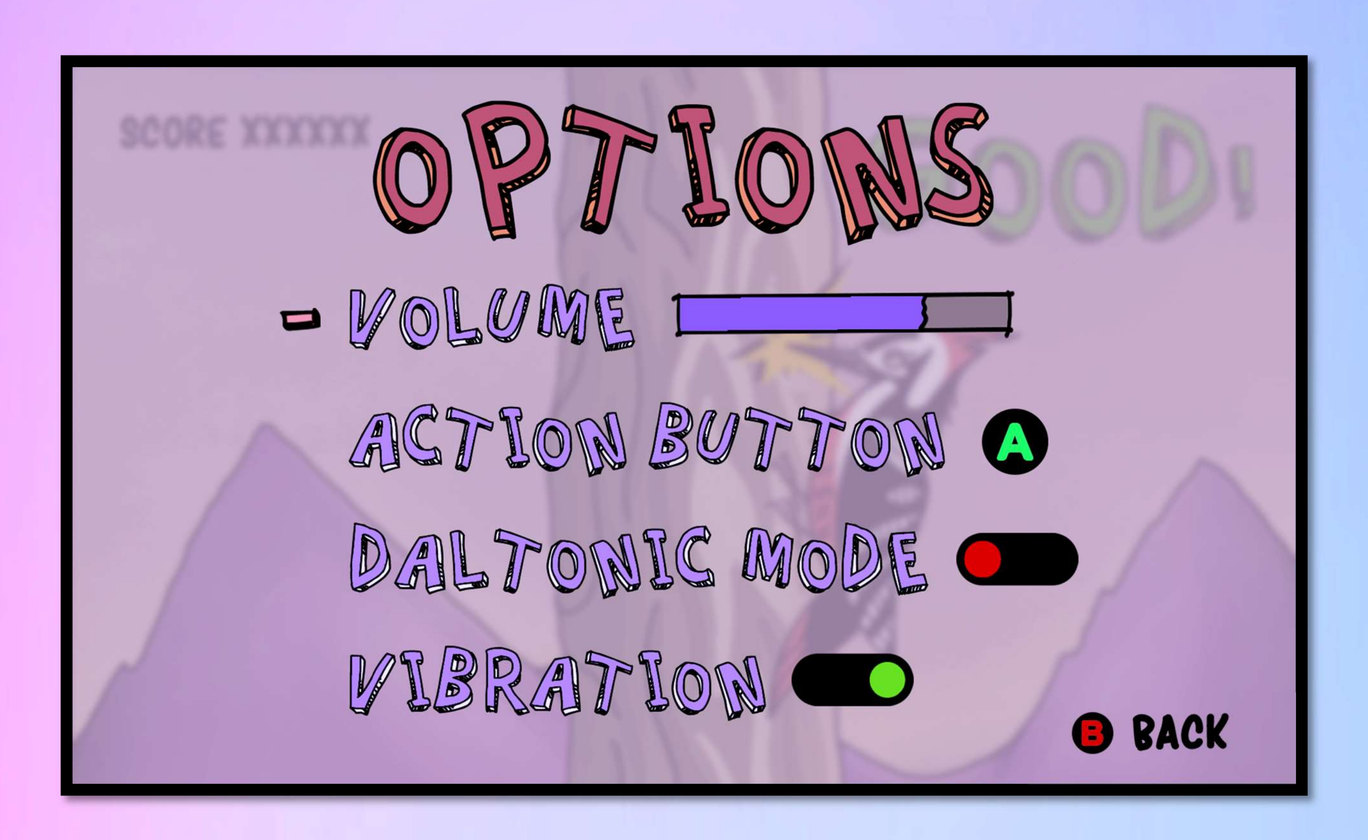
2. **Opción "*Resume*".**
   Permite continuar la partida.

3. **Opción "*Retry*".**
   Permite reiniciar el nivel sin reproducir las cinemáticas.

4. **Opción "*Options*".**
   Permite entrar en el menú de ajustes.

5. **Opción "*Exit*".**
   Permite salir al menú principal.

## ▪ MENÚ DE OPCIONES




### ▪ ELEMENTOS DEL MENÚ DE OPCIONES

1. **Título del menú.**
   "*Options*".

2. **Opción "*Volume*".**
   Permite ajustar el volumen general del juego.

3. **Opción "*Action Button*".**
   Permite ajustar el botón de acción ("*A*" por defecto).

4. **Opción "*Daltonic Mode*".**
   Permite activar el filtro de color para daltónicos.

5. **Opción "*Vibration*".**
   Permite activar o desactivar la vibración del mando.

# APARTADO VISUAL

## ▪ TEMÁTICA VISUAL

Como ya se explicó en la descripción del videojuego, todos los elementos (personajes y escenarios) van a ser 2D siguiendo un estilo "*cartoon*".

Predominará el uso de colores planos y trazados gruesos, siguiendo así el estilo de muchos referentes mostrados anteriormente.

## ▪ PERSONAJES

### ▪ PERSONAJES JUGABLES

1. **Minijuego pájaro hambriento**



- ***NPC's***

En la escena también aparecerán otros personajes no controlables (*NPC's*):

1. **Minijuego pájaro hambriento**



**Caracol**

Personaje decorativo. Aparecerá durante la primera cinemática.



**Gusano**

Personaje interactivo. Es el alimento que intenta conseguir el pájaro cuando pica.

Aparecerá y desaparecerá al ritmo de la música para tentar al personaje.

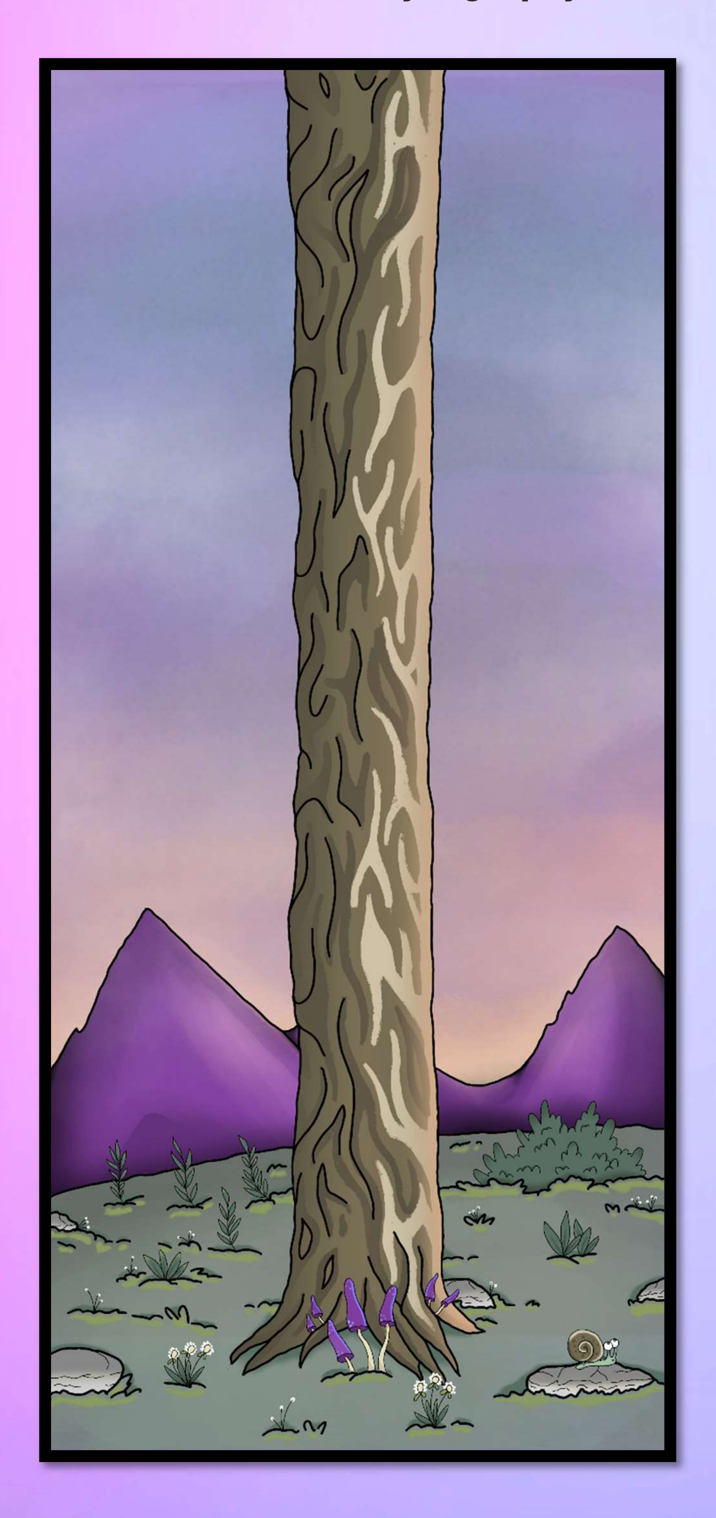
**NOTA:** Apartado “Localizaciones” en la siguiente página.

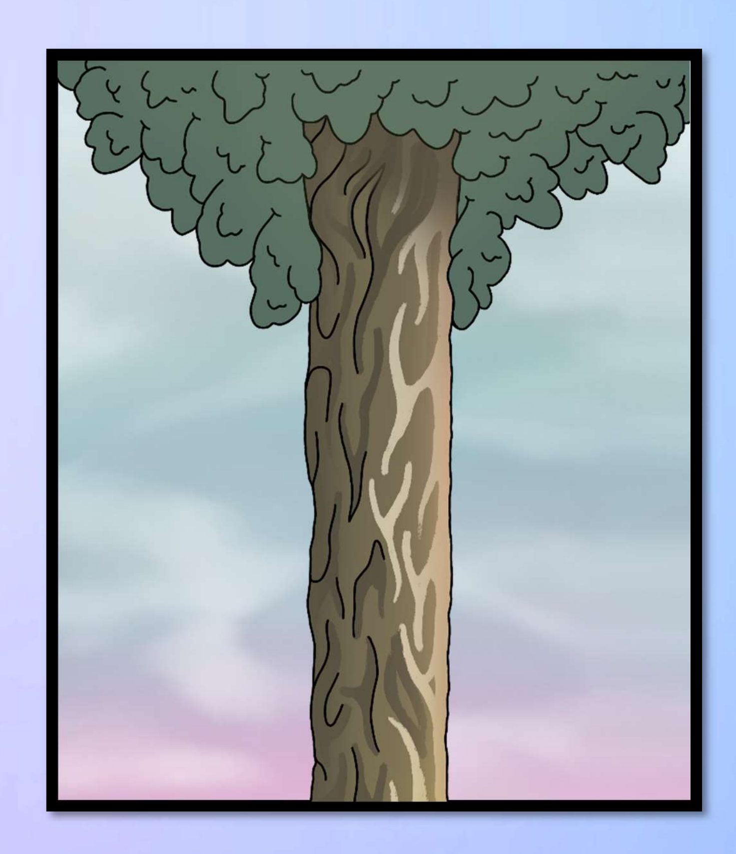
- # LOCALIZACIONES

  - ## ESCENAS (COMPOSICIONES)

A continuación aparecerán listados los diferentes escenarios del título:

1. **Minijuego pájaro hambriento**





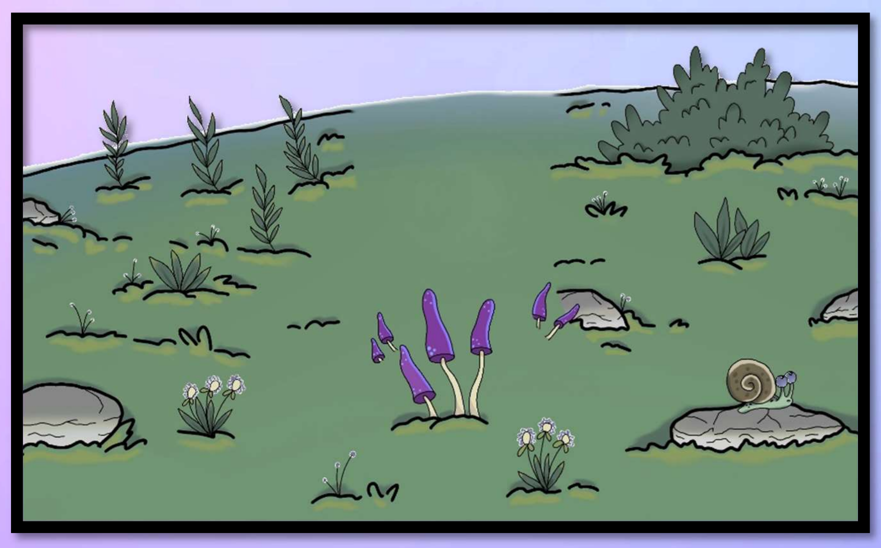
## ▪ ELEMENTOS DE LAS COMPOSICIONES

Estos son los diferentes elementos que compondrán cada escenario:

1. **Minijuego pájaro hambriento**

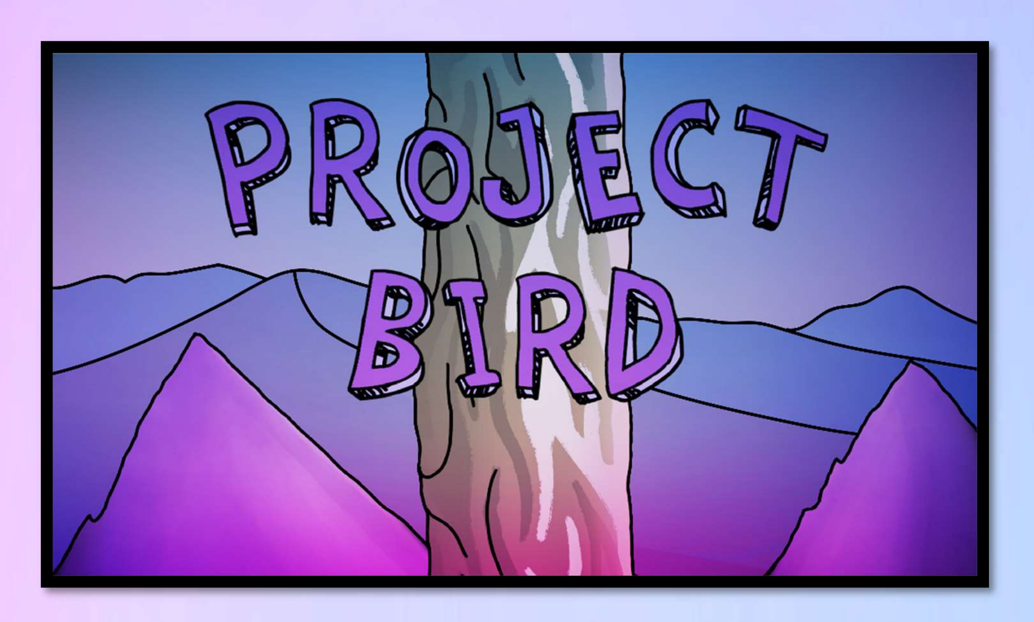






## ▪ FONDOS DE MENÚS

### ▪ PANTALLA DE INICIO




Fondo decorativo con logo del videojuego. En el futuro se sustituirá por una animación que vaya cambiando el color de la escena.

### ▪ MENÚ PRINCIPAL



Fondo decorativo. Irán cambiando las tonalidades cada vez que entremos al menú principal.

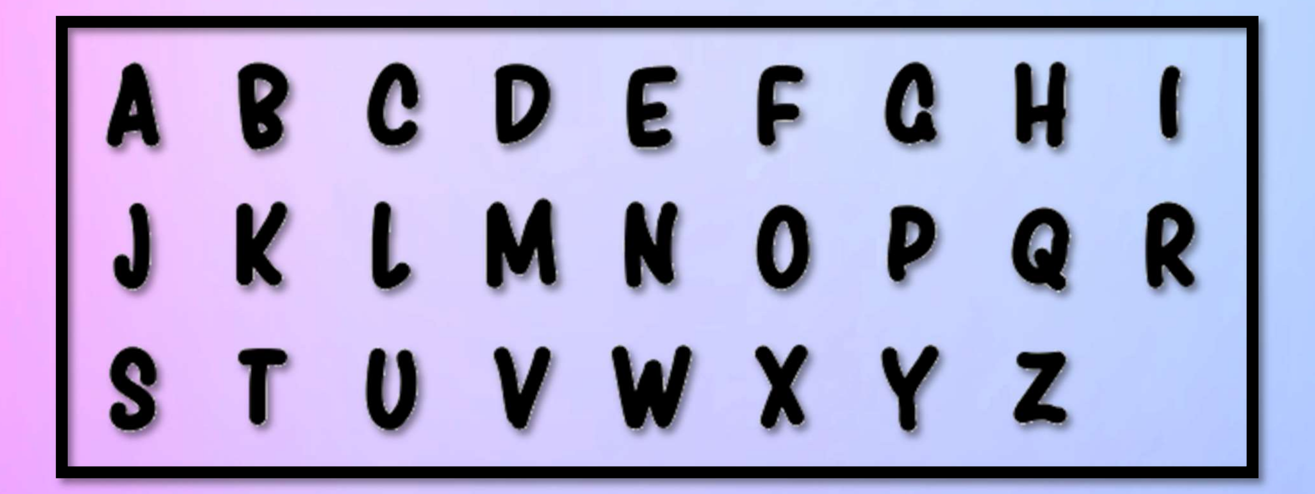
## ▪ TIPOGRAFÍAS EMPLEADAS

### ▪ CARTOON RELIEF




Fuente provisional. Empleada para el logo y textos decorativos.

### ▪ PINGWING




Fuente estándar para textos y números.

### ▪ BAHNSCHRIFT

abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
0123456789 (!#$%&/.|*`@´,?:;)

Fuente para documentación (textos, diapositivas, videos, etc.).

## ▪ VIDEOS Y CINEMÁTICAS

### ▪ INTRO SUNLIGHT STUDIO

Cuando inicie el videojuego, aparecerá una breve animación mostrando el logo de *Sunlight Studio*:











*Animación logo Sunlight Studio*

## ▪ CINEMÁTICAS DEL NIVEL

Como se explica en el apartado "Historia(s)", cada nivel tendrá dos cinemáticas.

## ▪ CRÉDITOS

Tras terminar todos los niveles, aparecerá un video con un mensaje de agradecimiento y los créditos del título.

**NOTA:** Apartado "Apartado sonoro" en la siguiente página.

# APARTADO SONORO

El apartado más destacable de este videojuego es el musical, siendo mucho más importante que otros tales como el estético o narrativo, ya que es el encargado de delimitar la dificultad, el ritmo del juego y la capacidad de rejugabilidad.

## ▪ ANÁLISIS MINIJUEGOS *RHYTHM PARADISE*

Para desarrollar correctamente un minijuego musical, es necesario analizar otros videojuegos con conceptos similares para evitar cometer ciertos errores.

En este caso, el referente principal es "*Rhythm Paradise*". Todos sus temas están hechos para encajar perfectamente con el minijuego en cuestión, además de incrementar la dificultad de forma progresiva para que el jugador entienda poco a poco la mecánica del juego y sienta una mejora real de sus capacidades.




*Minijuego de Rhythm Paradise*

*"Rhythm Paradise"* es consciente de que emplea melodías complejas. Superpone diferentes ritmos e instrumentos para acabar obteniendo una composición completa.

Esto se traduce en una progresión de dificultad interna, propia de cada nivel.

Para no confundir al jugador durante el transcurso de un minijuego, se emplea un único tipo de sonido con el que el jugador interactúa.

Vamos a explicar esto a través de dos minijuegos de "*Rhythm Paradise*":

1. **Minijuego 1 – Golf**






YouTube Video

Miremos el *gameplay.* Como hemos dicho antes, el jugador sólo responderá a un tipo de sonido, independientemente de cómo se complique la composición o melodía durante el transcurso del nivel.

En este caso, el jugador pulsa el botón de acción cuando el mono hace un ruido y el personaje golpea la pelota de golf. La música acompaña, pero podríamos eliminar la mayor parte de la melodía y el minijuego seguiría siendo jugable.

Esto es exactamente lo que buscamos para *Project Bird*, una forma especial de que el *gameplay* dependa de un ritmo, no de una canción en concreto.



*Momento en el que el mono hace un ruido y suena el golpe del palo de golf.*
*Momento en el que el jugador presiona el botón de acción.*

Más adelante aparece el gorila, que lanza las pelotas de golf más rápido y requiere de unos mejores reflejos por parte del jugador para acertar.

Combinando este tipo de sonidos se suma dificultad.

Eliminando animaciones y cinemáticas, el tema dura exactamente 1,5 minutos. Es el tiempo ideal para disfrutar un nivel y que la melodía pueda complicarse en términos de dificultad y calidad musical.

## 2. Minijuego 2 – Avionetas y bádminton

Este segundo ejemplo sirve de refuerzo. Funciona exactamente igual que el anterior, pero con una melodía y ritmo diferente.





YouTube Video

Miremos el *gameplay.* Al igual que antes, el jugador presiona el botón de acción cuando se escucha un ruido en concreto y la pelota de bádminton se acerca.

La melodía se va complicando instrumental y rítmicamente, pero el jugador seguirá teniendo que responder únicamente al estímulo que le presentaron al principio.

La duración del tema es corta. En este caso 1 minuto y 41 segundos.

# ▪ MÚSICA (TEMA PRINCIPAL)

Para este primer minijuego, contaremos únicamente con un tipo de sonido con el que el jugador interactuará, por lo que aumentaremos la dificultad progresivamente acelerando su ritmo.

Para lograr esto, lo más importante de todo es crear temas/obras/melodías cortas y repetibles, que puedan reproducirse una y otra vez para desarrollar la historia.

También será necesario acelerar estas partes para aportar altibajos e incrementar la dificultad.

En resumen, la música deberá cumplir los siguientes requisitos:

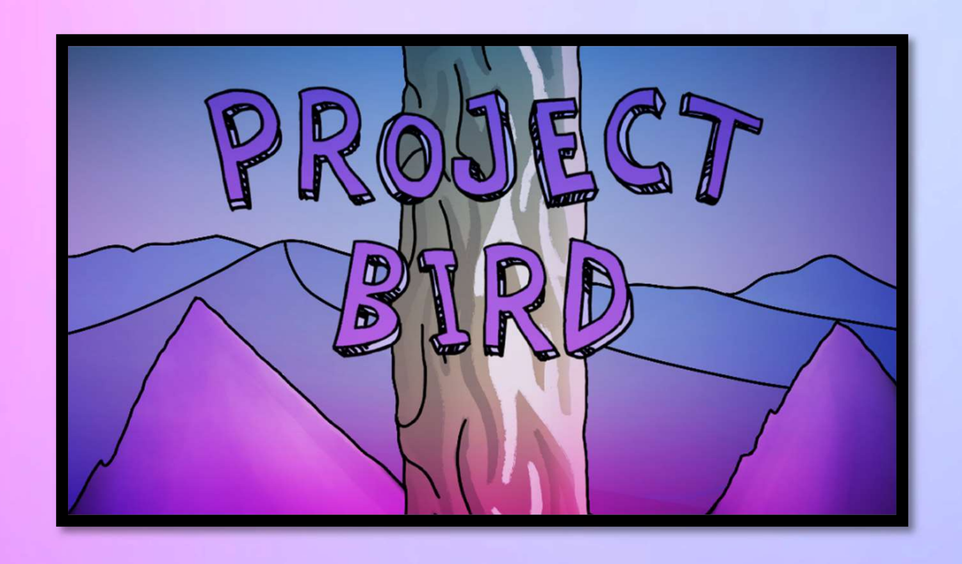
1. La duración del tema deberá ser breve. No debe durar más de 1 minuto y 45 segundos.

2. El sonido al que tendrá que responder el jugador debe ser muy característico y completamente paralelo al tema en cuestión.

3. Para mantener cierta concordancia estética, las melodías deben ser compuestas teniendo en cuenta la ambientación del minijuego.

4. Se sumará dificultad acelerando el ritmo de la melodía. En un futuro, se podrá emplear más de un sonido de acción para hacer más orgánico este proceso.

La canción que tenemos compuesta actualmente para realizar pruebas de jugabilidad y funcionamiento es la siguiente:






YouTube Video

# TECNOLOGÍAS EN USO Y REQUERIMIENTOS TÉCNICOS

## ▪ SOFTWARE EMPLEADO

El videojuego va a ser programado completamente usando el motor Unity.

Por otro lado, necesitaremos emplear otras herramientas tales como *ProCreate, Adobe Phototoshop, Finale, FL Studio, Adobe After Effects y Adobe Premiere.*















## ▪ REQUISITOS MÍNIMOS (PC)

Los requisitos mínimos recomendados provisionales serán los siguientes:

- Procesador Dual Core (~2 GHz) de 64 bits.
- 2 / 4 GB RAM.
- Nvidia GTX 650 1GB / AMD RX Vega 3 (APU).

# [ANEXOS] DIAPOSITIVAS DE ESPECIFICACIÓN

- **ANEXO 1 – PITCH DE PROPUESTA**

Sunlight STUDIO

GAME DESIGN #5

INTERFACE, HUD & FLOW CHARTS

JUAN MANUEL RUIZ CULIÁÑEZ
SUNLIGHT STUDIO

@JUANMACOLIBRI
@SUNLIGHTABP

-FEBRUARY 2021-

CODENAME :

PROJECT BIRD



YouTube Video

# PREVIOUS GAMES / WORKS

2019 — 2020 — 2021







HEY SAILOR! (CPC 464 GAME)




RUNNINJA (CPC 464 GAME)



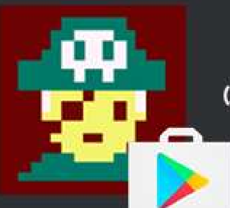

MOONLIGHT ENGINE
(OWN GAME ENGINE)



CYBORGEDDON (PC GAME)




KALINKA (PC GAME)



HEY SAILOR!
(ANDROID PORT)

PROJECT BIRD (?)

PROJECT BIRD (?)

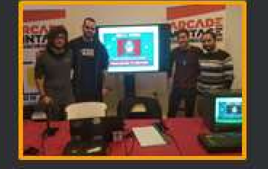
AWARDS




BEST ACTION GAME
(AMSTRAD ETERNO)




5TH BEST GAME
(CPC RETRO DEV)



ARCADE VINTAGE TALKS

# GAME PROPOSAL

## 1. VIDEOGAME TYPE

- RHYTHM GAME.
- THEMED MICRO-STORIES.
- DIFFERENT LEVELS (MICRO-STORIES).
- FOR ALL AUDIENCES.

## 2. ESTHETICS

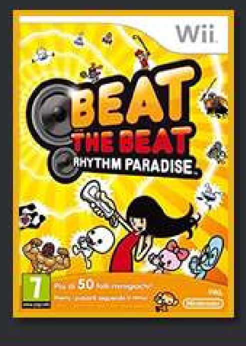
- CARTOON STYLE.
- FRIENDLY APPEARANCE.



## 3. OBJECTIVE

- CREATE A SIMPLE GAME.
- PROFESSIONAL AAA FINISH.
- MUSIC AND ARTWORK SPECIFICALLY DESIGNED FOR EACH MINIGAME.

## 4. REFERENCES (GAMES, VISUALS & MUSIC).













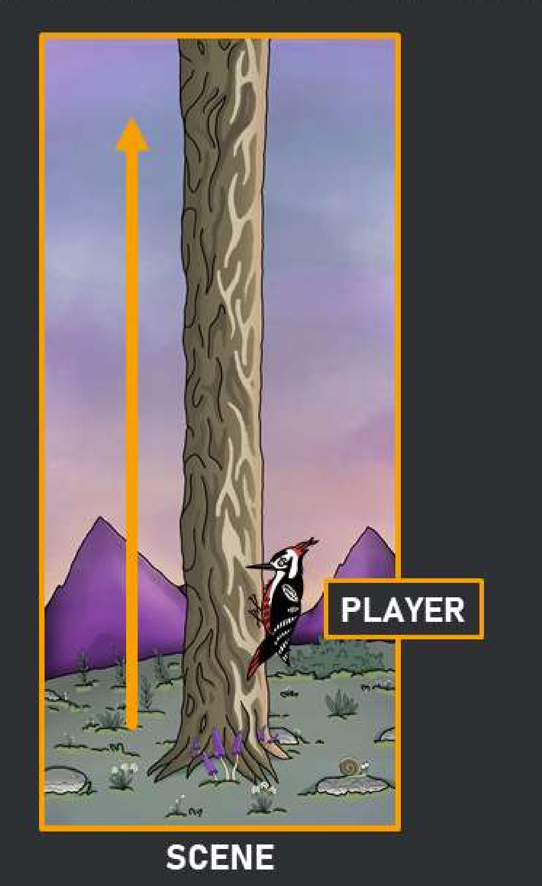








# LEVEL EXAMPLE




SCENE



CHARACTER



GAMEPLAY EXAMPLE

## BONUS INFO

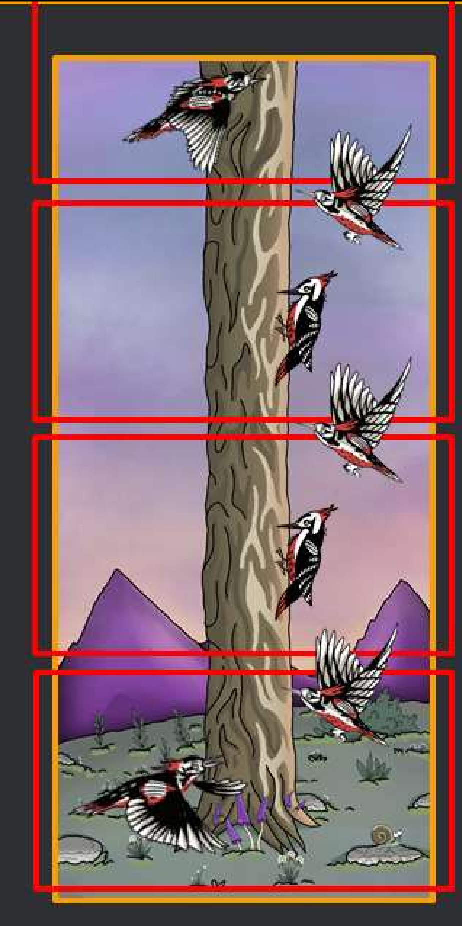
- SINGLE-BUTTON CONTROL.
- PLAYER MARKS BIRD'S PECKING RHYTHM.
- DURATION: 1,5 ~ 2 MINUTES.
- "FUNKY" MUSIC GENRE.
- RECYCLABLE ASSETS
- EXTREME RE-PLAYABILITY. (SCORES & LEADERBOARDS).

# LEVEL EXAMPLE



**STATIC CAMERA, NOT CONTROLLABLE**

- VISUAL EFFECTS LIKE PARTICLES, PARALLAX, ETC.



**FINAL SCENE**
- ENDING ANIMATION.
- NON-PLAYABLE SCENE

**TRANSITION** (NON-PLAYABLE)

**SCENE 3**
- GAMEPLAY.
- BIRD'S PECKING.

**TRANSITION** (NON-PLAYABLE)

**SCENE 2**
- GAMEPLAY.
- BIRD'S PECKING.

**TRANSITION** (NON-PLAYABLE)

**SCENE 1**
- OPENING ANIMATION.
- NON-PLAYABLE SCENE.
- MUSIC STARTS.

**THE CAMERA FOLLOWS CHARACTER'S MOVEMENT AUTOMATICALLY (SCROLL)**

# GAME'S VALUE AS A PRODUCT

## 1. KEY POINTS

- ALL AUDIENCES. (PEGI 3).

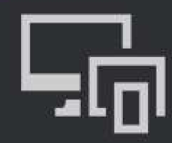



- EASY TO PLAY.



- EXTREME RE-PLAYABILITY.
  (SCORES & LEADERBOARDS).

- COMPATIBLE WITH ALL PLATFORMS.





- FAST DEVELOPMENT. (< 1 YEAR)
  LITE VERSION LAUNCH.

## 2. BUSINESS MODEL

- MOBILE PREMIERE LAUNCH.




- PERIODICAL UPDATES WITH NEW CONTENT. (MORE LEVELS).




- DEFINITIVE EDITION FOR CONSOLES AND PC. (INCLUDING ALL CONTENT).








## 3. ADAPTATIVE STRATEGY

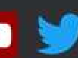
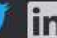
- OPEN TO RE-ADJUST SOME BUSINESS STRATEGIES.








# THANKS!

- SUNLIGHT STUDIO (@SUNLIGHTABP) 
- JUAN MANUEL RUIZ (@JUANMACOLIBRI)  
- JUANMACOLIBRIDEV@GMAIL.COM 



## ▪ ANEXO 2 – PITCH DE CÁMARA Y CONTROLES

UNIVERSAL U ARTS SCHOOL – SINCE 1994 –

PRÁCTICA 4
GAME DESIGN:
CÁMARA Y CONTROLES

JUAN MANUEL RUIZ CULIÁÑEZ @JUANMACOLIBRI
(CURSO 2020 - 2021)
TUTORES: RAMÓN NAFRIA Y VÍCTOR ESCRIVÁ

PROJECT BIRD



YouTube Video

# PROCEDIMIENTO

## 1. PRESENTACIÓN DEL JUEGO → 2. CÁMARA → 3. CONTROL DEL JUEGO

**1. PRESENTACIÓN DEL JUEGO**

- TIPO DE VIDEOJUEGO (GÉNERO).
- ESTÉTICA.
- OBJETIVO.
- REFERENTES.

**2. CÁMARA**

- DEFINICIÓN.
- EJEMPLOS.
- TIPOS DE SITUACIONES.
- DISECCIÓN DE UN NIVEL.
- EFECTOS DE CÁMARA.

**3. CONTROL DEL JUEGO**

- LISTA DE ACCIONES.
- LISTA DE CONTROLES (PARA CADA PLATAFORMA).
- PECULIARIDADES DEL CONTROL (PARA CADA PLATAFORMA).
- MEDIDAS A TOMAR (SOLUCIONES).

# PRESENTACIÓN DEL JUEGO

## 1. TIPO DE VIDEOJUEGO

- JUEGO RÍTMICO.
- MINI-HISTORIAS TEMÁTICAS.
- DIFERENTES NIVELES (MINI-HISTORIAS).
- JUEGO PARA TODOS LOS PÚBLICOS.

## 2. ESTÉTICA

- CARTOON.
- AMIGABLE.



## 3. OBJETIVO

- CREAR UN VIDEOJUEGO SENCILLO, PERO CON ACABADO PROFESIONAL.
- CREACIÓN DE PERSONAJES E HISTORIAS DIFERENTES PARA CADA MINIJUEGO.
- MÚSICA DISEÑADA Y COMPUESTA ESPECÍFICAMENTE PARA CADA MINIJUEGO.

## 4. REFERENTES (VIDEOJUEGOS, ANIMACIÓN Y MÚSICA).













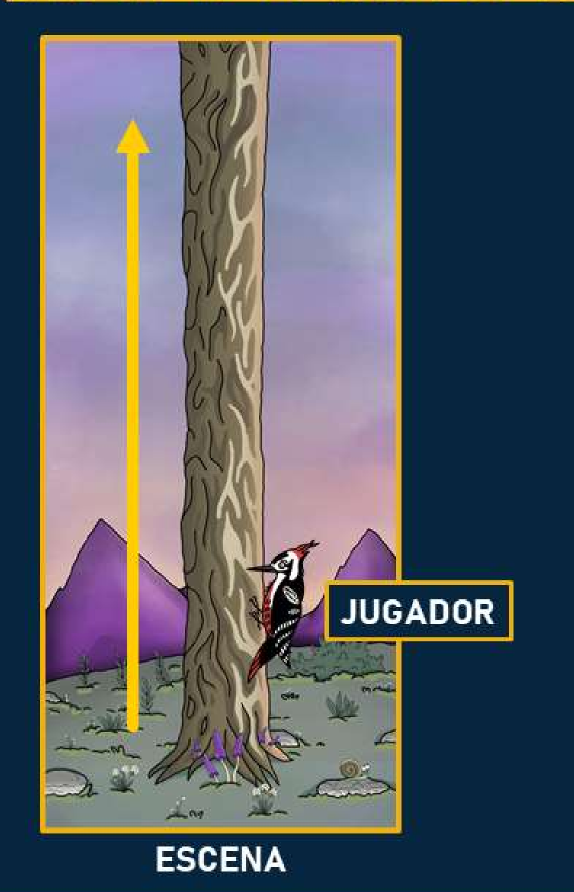








# NIVEL EN DESARROLLO




ESCENA



PERSONAJE



EJEMPLO GAMEPLAY

## DATOS EXTRA

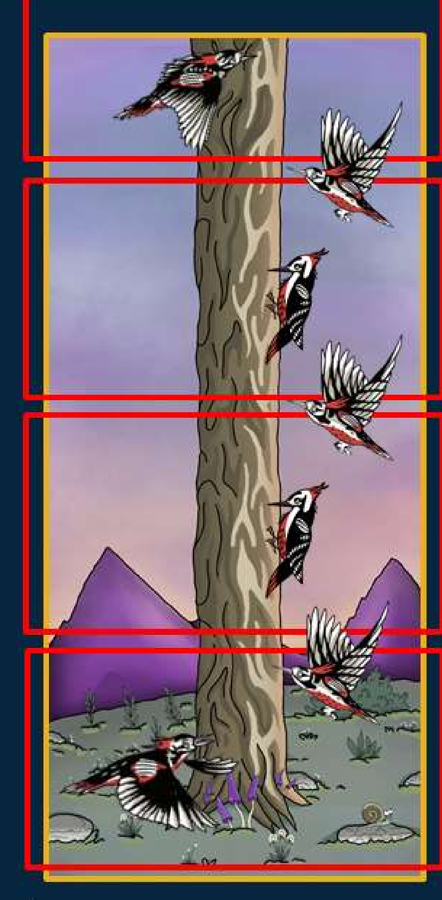
- CONTROL MEDIANTE UN ÚNICO BOTÓN.
- PICOTEO DEL PÁJARO COMO ACCIÓN A CONTROLAR.
- DURACIÓN: 1,5 ~ 2 MINUTOS.
- MÚSICA "FUNKY".
- PERSONAJE RECICLABLE PARA MÁS NIVELES.
- REJUGABILIDAD EXTREMA (PUNTUACIONES Y CALIFICACIONES).

# CÁMARA



**CÁMARA ESTÁTICA, NO CONTROLADA POR EL JUGADOR**

- ADICIÓN DE EFECTO PARALLAX.



**ESCENA FINAL**
- ANIMACIÓN FINAL.
- ZONA NO JUGABLE.

**TRANSICIÓN** (NO JUGABLE)

**ESCENA 3**
- GAMEPLAY.
- PICOTEO DEL PÁJARO.

**TRANSICIÓN** (NO JUGABLE)

**ESCENA 2**
- GAMEPLAY.
- PICOTEO DEL PÁJARO.

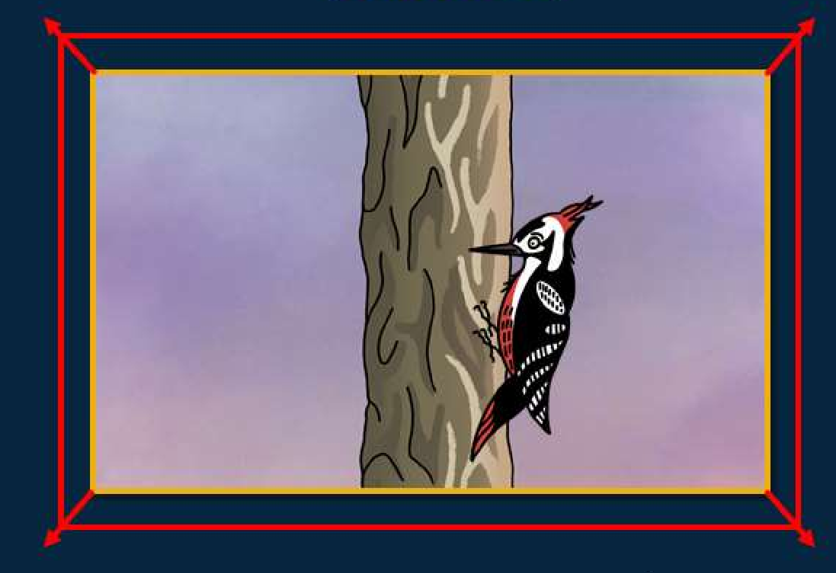
**TRANSICIÓN** (NO JUGABLE)

**ESCENA 1**
- ANIMACIÓN INTRODUCTORIA.
- ZONA NO JUGABLE.
- COMIENZO DE LA MÚSICA.

**LA CÁMARA SE DESPLAZA JUNTO AL PERSONAJE AUTOMÁTICAMENTE (SCROLL)**

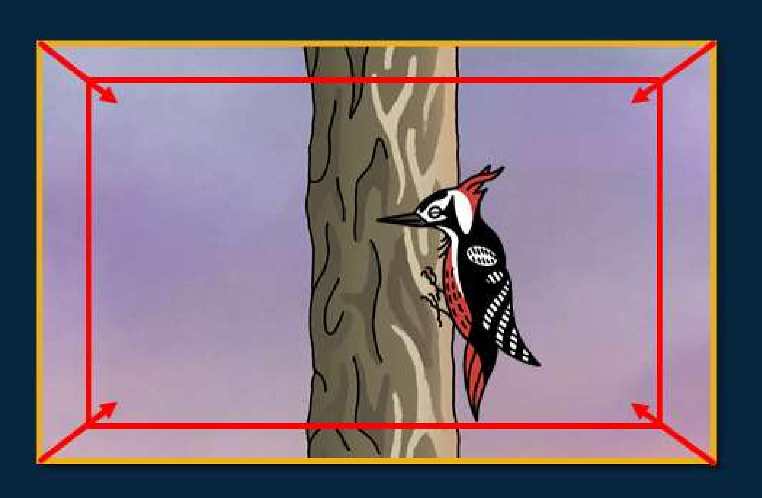
# EFECTOS DE CÁMARA

## EFECTO #1



**ZOOM OUT (1% ~ 2%)** EN FUNCIÓN DE
LOS "BPM" (BEATS PER MINUTE)
DE LA CANCIÓN.

## EFECTO #2



**ZOOM IN (2%)** CADA VEZ QUE EL
JUGADOR PULSE EL BOTÓN
(CADA VEZ QUE PICA EL PÁJARO).

# CONTROLES (MANDO)

MENÚS
ACCIÓN

OPCIONES
PAUSA
ACCIÓN (PÁJARO PICA)
(CUALQUIERA DE LOS 4 BOTONES, CONFIGURABLE)
Y
X
B
A
NAVEGACIÓN EN MENÚ
ACEPTAR
CANCELAR

*NOTA*

cs = centésimas de segundo.

ms = milésimas de segundo.

## DETALLES ADICIONALES

- LATENCIA MANDO XBOX ONE.
  (+17,4 ms)
- LATENCIA MANDO PLAYSTATION 4.
  (+13,3 ms)

## MEDIDAS A TOMAR

MARGEN BASE = ± 12cs

- CORRECCIÓN INPUT XBOX ONE.
  (BEAT-TIME ± 12 cs ± 17,4 ms) ⟶ ± 13,74 cs
- CORRECCIÓN INPUT PLAYSTATION 4.
  (BEAT-TIME ± 12 cs ± 13,3 ms) ⟶ ± 13,33 cs




***NOTA***

**cs = centésimas de segundo.**

**ms = milésimas de segundo.**

## DETALLES ADICIONALES

- LATENCIA TECLADO GENÉRICO (USB). (+55 ms)
- LATENCIA TECLADO GAMING (USB). (+25 ms)

## MEDIDAS A TOMAR

MARGEN BASE = ± 12cs

- CORRECCIÓN TECLADO GENÉRICO. (BEAT-TIME ± 12 cs ± 55 ms) ⟶ ± 17,50 cs
- CORRECCIÓN TECLADO GAMING. (BEAT-TIME ± 12 cs ± 25 ms) ⟶ ± 14,50 cs

# ¡MUCHAS GRACIAS!

- UNIVERSAL ARTS SCHOOL (@UASCHOOL)
- JUAN MANUEL RUIZ (@JUANMACOLIBRI)
- JUANMACOLIBRIDEV@GMAIL.COM



## ▪ ANEXO 3 – PITCH DE INTERFAZ

Sunlight STUDIO

GAME DESIGN #5

INTERFACE, HUD & FLOW CHARTS

JUAN MANUEL RUIZ CULIÁÑEZ @JUANMACOLIBRI

SUNLIGHT STUDIO @SUNLIGHTABP

-FEBRUARY 2021-

CODENAME :

PROJECT BIRD



YouTube Video

# INDEX



# GAME INTERFACE

*GAME UNDER DEVELOPMENT. VISUALS AND INTERFACES ARE CONCEPTUAL AND MAY CHANGE IN THE FINAL PRODUCT.




## TITLE SCREEN




- INFORMATION CHANNELS:
  - TEXT. (EXPLICIT).
  - SOUND WHEN PRESSING BUTTON. (IMPLICIT).

## MAIN MENU




- INFORMATION CHANNELS:
  - ICONS & TEXT. (EXPLICIT).
  - SOUND WHEN PRESSING BUTTON. (IMPLICIT).

# GAME INTERFACE

*GAME UNDER DEVELOPMENT. VISUALS AND INTERFACES ARE PROVISIONAL AND MAY CHANGE IN THE FINAL PRODUCT.




## IN-GAME







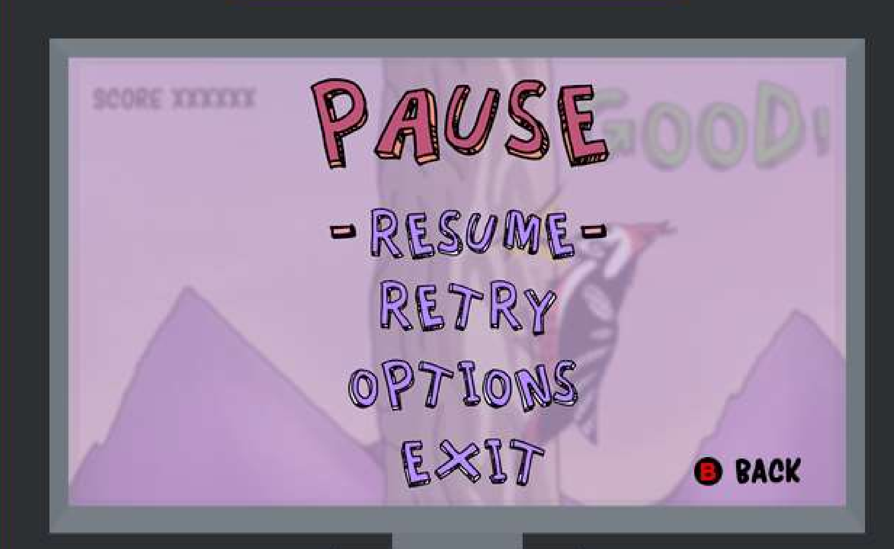

- INFORMATION CHANNELS:
  - TEXT. (EXPLICIT).
  - REAL-TIME ANIMATIONS. (EXPLICIT).
  - MUSIC BASS RHYTHM. (IMPLICIT).
  - SOUND WHEN BIRD PECKS. (IMPLICIT).
  - FEEDBACK SOUNDS. (IMPLICIT).

# GAME INTERFACE

*GAME UNDER DEVELOPMENT. VISUALS AND INTERFACES ARE CONCEPTUAL AND MAY CHANGE IN THE FINAL PRODUCT.



## PAUSE SCREEN




▸ INFORMATION CHANNELS:

- TEXT. (EXPLICIT).
- SOUND WHEN PRESSING BUTTON. (IMPLICIT).

## OPTIONS SCREEN




▸ INFORMATION CHANNELS:

- ICONS & TEXT. (EXPLICIT).
- SOUND WHEN PRESSING BUTTON. (IMPLICIT).

# GAME STATES

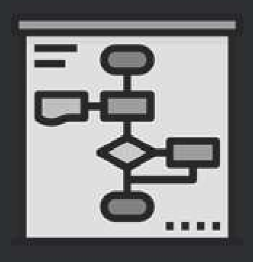
*GAME UNDER DEVELOPMENT. VISUALS AND INTERFACES ARE CONCEPTUAL AND MAY CHANGE IN THE FINAL PRODUCT.



## WHY DO WE NEED A GAME STATES FLOW CHART ?

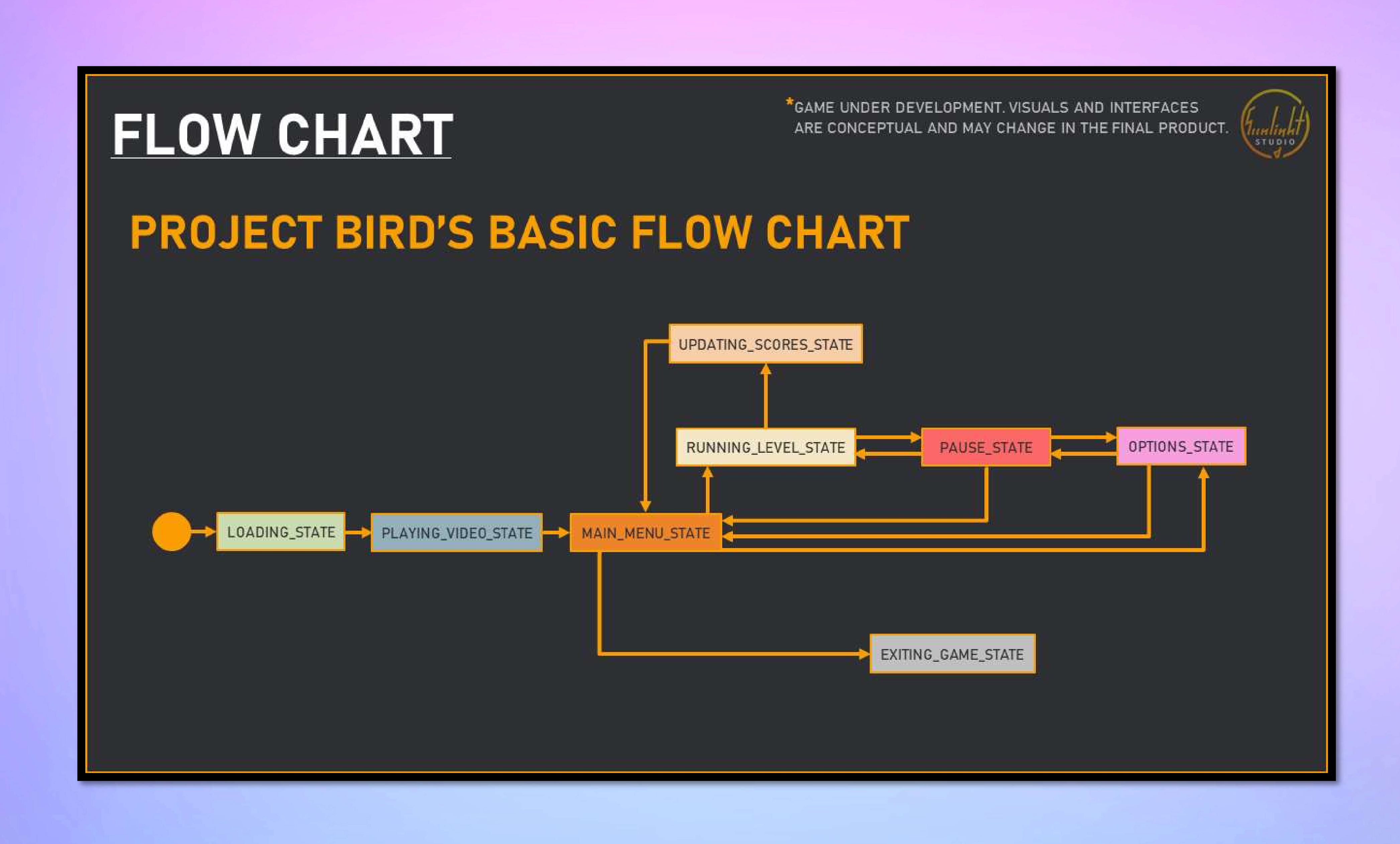
- GAMES HAVE DIFFERENT NECESITIES.
- EACH GAME WORKS DIFFERENTLY.
- EACH GAME HAS DIFFERENT STATES.
- VERY USEFUL DURING TECH IMPLEMENTATION.





### PROJECT BIRD GAME STATES

- PLAYING_VIDEO_STATE
- MAIN_MENU_STATE
- LOADING_STATE
- RUNNING_LEVEL_STATE
- UPDATING_SCORES_STATE
- PAUSE_STATE
- OPTIONS_STATE
- EXITING_GAME_STATE

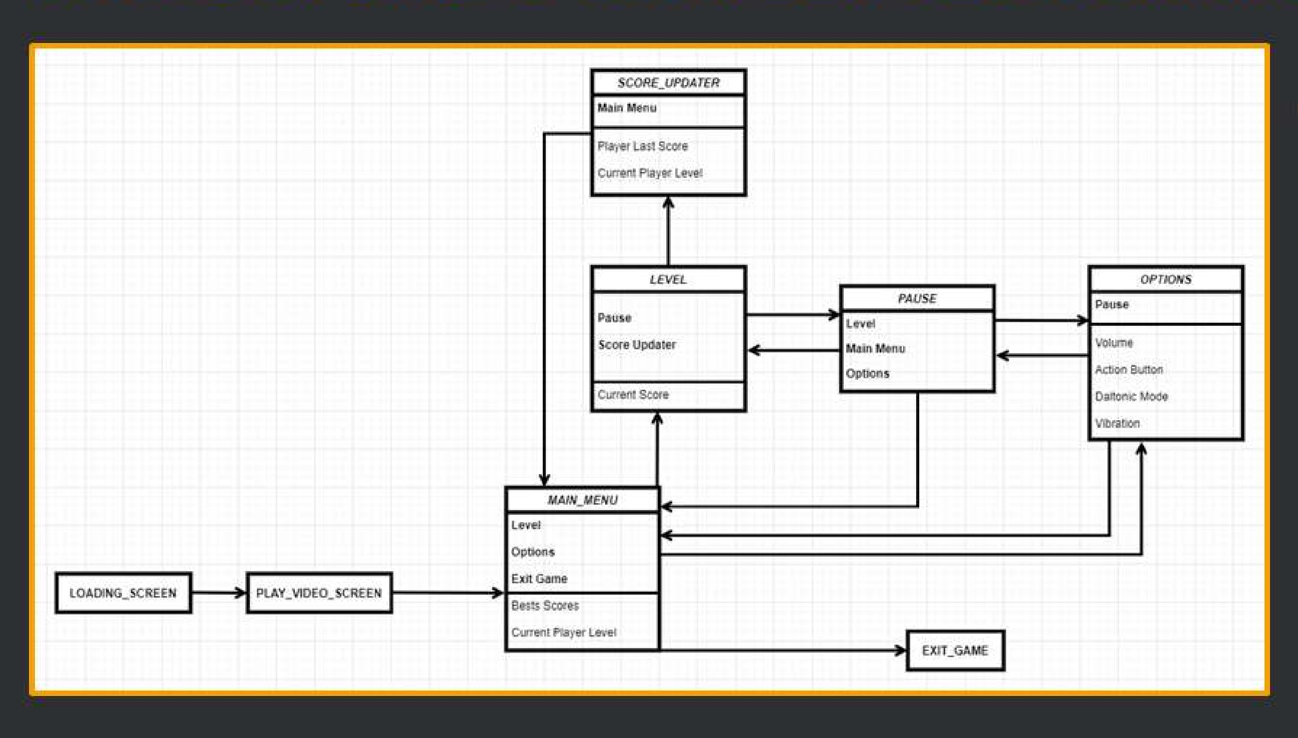

FLOW CHART
*GAME UNDER DEVELOPMENT. VISUALS AND INTERFACES
ARE CONCEPTUAL AND MAY CHANGE IN THE FINAL PRODUCT.
Sunlight
STUDIO
PROJECT BIRD'S BASIC FLOW CHART
UPDATING_SCORES_STATE
RUNNING_LEVEL_STATE
PAUSE_STATE
OPTIONS_STATE
LOADING_STATE
PLAYING_VIDEO_STATE
MAIN_MENU_STATE
EXITING_GAME_STATE

# FLOW CHART

*GAME UNDER DEVELOPMENT. VISUALS AND INTERFACES ARE CONCEPTUAL AND MAY CHANGE IN THE FINAL PRODUCT.

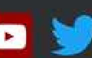



## PROJECT BIRD'S COMPLEX FLOW CHART




- OTHER HELPFUL DIAGRAMS:
  - CLASS DIAGRAM. (PROGRAMMING).
  - ECS DIAGRAM. (ENTITY-COMPONENT-SYSTEM).
  - IMPLEMENTATION ORDER DIAGRAM. (PLANIFICATION).

# THANKS!

- SUNLIGHT STUDIO (@SUNLIGHTABP) 
- JUAN MANUEL RUIZ (@JUANMACOLIBRI)  

